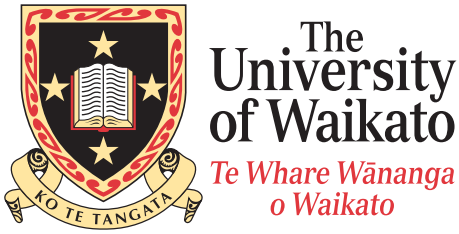
Department of Computer Science



The University of Waikato
Te Whare Wānanga o Waikato

Hamilton, NewZealand

Effective Linear-Time Feature Selection

Nripendra Pradhananga

This thesis is submitted in partial fulfilment of the requirements for the degree of Master of Science at The University of Waikato.

July 2007

Abstract

The classification learning task requires selection of a subset of features to represent patterns to be classified. This is because the performance of the classifier and the cost of classification are sensitive to the choice of the features used to construct the classifier. Exhaustive search is impractical since it searches every possible combination of features. The runtime of heuristic and random searches are better but the problem still persists when dealing with high-dimensional datasets.

We investigate a heuristic, forward, wrapper-based approach, called *Linear Sequential Selection*, which limits the search space at each iteration of the feature selection process. We introduce randomization in the search space. The algorithm is called *Randomized Linear Sequential Selection*. Our experiments demonstrate that both methods are faster, find smaller subsets and can even increase the classification accuracy.

We also explore the idea of ensemble learning. We have proposed two ensemble creation methods, *Feature Selection Ensemble* and *Random Feature Ensemble*. Both methods apply a feature selection algorithm to create individual classifiers of the ensemble. Our experiments have shown that both methods work well with high-dimensional data.

Acknowledgements

First and foremost I would like to thank my supervisors Dr Bernhard Pfahringer and Dr Mark A. Hall for their patience and continued guidance to making this thesis a success. Thanks for repeatedly explaining the problem in the early stage, your encouragement, and going through my final thesis in detail though you were busy.

I am also grateful to Kay for reading my thesis and giving me writing advices. I thank Asraf for technical assistance and helpful comments. Many thanks to the people of the ML group for being such a great team to work with.

Special thanks go to my cousin Jay for everything.

Finally, thank you Mum and Dad for supporting me financially and in many other ways. It would simply not be possible for me to go this far without your unconditional support

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Machine learning is an area of artificial intelligence that has grown tremendously in the last two decades due to its utility in real world applications such as speech recognition, visual recognition, text classification and computer vision. Machine learning on the whole is concerned with concept learning. One style of concept learning is classification learning. In classification learning, a learning algorithm is typically presented with a set of instances, where each instance is described by a fixed number of features along with a label that denotes its class. The learning algorithm then outputs a concept description that represents underlying patterns in the data.

In machine learning, the learning algorithms have to deal with large amounts of data. Usually, the data contains thousands or tens of thousands of instances and each instance is represented by hundreds to many thousands of features. For example, in gene expression microarray data, the features represent gene expression coefficients corresponding to the abundance of mRNA in a sample (e.g. tissue biopsy), for a number of patients. Although there are usually very few examples (patients) (often less than 100) for training and testing, the number of features in the raw data ranges from 6,000 to 60,000. A typical classification task is to separate healthy patients from cancer patients based on their gene expression "profile".

Theoretically, learning from many features should result in higher predictive accuracy. However, experimental evidence has shown that this is not always the case. Decision

tree learners, such as C4.5 [35], are known to degrade in performance when faced with many irrelevant features. Divide-and-conquer tree learners and separate-and-conquer rule learners exhibit similar effects when faced with irrelevant features. Similarly, instance-based learners are also very susceptible to irrelevant features. It has been shown that the number of training instances needed to produce a predetermined level of performance for instance-based learning increases exponentially with the number of irrelevant features present [28]. On the other hand, algorithms such as naive Bayes are robust with respect to irrelevant features. Their performance degrades very slowly when more irrelevant features are added. However, the performance of such algorithms degrade quickly by adding redundant features. Even if they are relevant to the concept. Furthermore, the presence of many features not only affects the predictive performance, but also the runtime of the learning algorithms. As the number of features increases, the runtime of the learning algorithm increases. The problem stated is also referred to as the "curse of dimensionality" [12].

Due to the aforementioned problems, many learning algorithms employ the principle of Occam's Razor [14] when building a model. This principle states that the explanation of any phenomenon should make as few assumptions as possible, eliminating those that make no difference in the observable predictions of the explanatory hypothesis or theory [40]. This bias often leads an algorithm to choose a small number of relatively predictive features over a large number of features that, taken in a proper combination, are fully predictive of the class label. Thus, the algorithms that reduce the amount of data and focus on the relevant features and instances are useful pre-processing methods for learning algorithms. These methods are known as feature selection methods.

Feature selection methods are fundamental to machine learning. Feature selection is defined as the selection of a subset of features such that the learning algorithm focuses only on those aspects of the data which are useful for analysis and future prediction. Based on the search strategy employed, feature selection methods can be placed into three broad categories: exhaustive search, heuristic search and random search. Exhaustive search, as the name suggests, searches every possible combination of features in the search space to

find the optimal subset of features. Exhaustive search is impractical for high-dimensional datasets since its time complexity, i.e. the number of feature subsets that need to be evaluated, is $O(2^n)$. It is obvious that exhaustive search always finds the optimal subset. Heuristic search must be used when there is a non-trivial number of features. The motivation behind the heuristic search is to find a near optimal, if not optimal, feature of subset in an acceptable amount of time. The time complexity of the heuristic search is $O(n^2)$. The random method produces feature subsets at random. The motivation behind the randomized search method is to avoid getting stuck in local minima and to capture interdependencies between the features. Furthermore, feature selection can be divided into two categories – filter-based and wrapper-based. In the filter-based approach, feature selection is performed independently of a learning algorithm. In the wrapper-based approach, a learning algorithm itself is used to measure the goodness of feature subsets. Wrapper-based approaches often outperform filter methods but are much slower. [24]

We propose a new search strategy, called *Linear Sequential Selection*. *Linear Sequential Selection* is a variant of forward selection that evaluates only the best k features for potential addition to the set of features at each iteration of the selection process, rather than all n remaining features. We also introduce randomization in *Linear Sequential Selection* to address the local minima problem and to improve classification accuracy. This algorithm is called *Randomized Linear Sequential Selection*. In *Randomized Linear Sequential Selection*, the m number of least favorable features in the best k features are replaced by randomly selecting features at each iteration of the selection process.

This thesis also explores the idea of an ensemble learning method based on the feature selection techniques described above in order to improve the classification accuracy of the learning algorithms. In ensemble learning, a set of classifiers are built from a single classifier, called the base classifier, by changing the training set or the input features. Then, each classifier votes to decide the final classification. Bagging [10] and Boosting [36] are two popular examples of ensemble learning. Both methods build different classifiers using a base learning algorithm by changing the distribution of the training set. Another approach for building multiple classifiers is to use different feature subsets. Current work

involves using randomization, i.e. randomly selecting features for each classifier. Examples include random forests [19] and MFS [6].

We propose two new ensemble creation methods that are based on the selection of particular subsets of features. The algorithms are called *Feature Selection Ensemble* and *Random Feature Ensemble*. Both algorithms are based on the idea of applying feature selection to find useful features from which individual classifiers are trained. For *Feature Selection Ensemble*, any known feature selection algorithm can be used. After constructing a classifier based on the result of feature selection, the selected features are removed from the training set and the process is repeated for the next classifier. *Random Feature Ensemble* directly applies *Randomized Linear Sequential Selection* to select the features for each classifier.

The main contribution of this thesis are algorithms for feature selection that, on average, considerably improve over the exhaustive search algorithms in terms of runtime. A further contribution is provided through new ensemble creation methods that show improvement in the prediction accuracy over single learning algorithms. Furthermore, these algorithms are simple, easy to implement, and have low storage cost.

1.1 Overview

The rest of the thesis is organized as follows. Chapter 2 provides background information on the topic of feature selection. Chapter 3 provides background information on the topic of ensemble learning. Chapter 4 introduces four new algorithms – two for feature selection and two for ensemble learning. Chapter 5 reviews the general methodology used to evaluate the performance of the algorithms. Chapter 6 provides an analysis of the experimental results obtained from the evaluation of the algorithms. Finally, Chapter 7 summarizes the contribution of this thesis and suggests avenues for future work.

Chapter 2

Feature Subset Selection

Feature selection algorithms are fundamental for machine learning. Feature selection is defined as the process of identifying and removing irrelevant and redundant features from the training data, so that the learning algorithm focuses only on those aspects of the training data useful for analysis and future prediction. Furthermore, the reduced dimensionality of the training data allows learning algorithms to operate faster. Feature subset selection is usually employed as a preprocessing step for a learning algorithm in order to boost performance. In other words, it can be viewed as a filter for the learning algorithm. Figure 2.1 illustrates the feature selection methods.

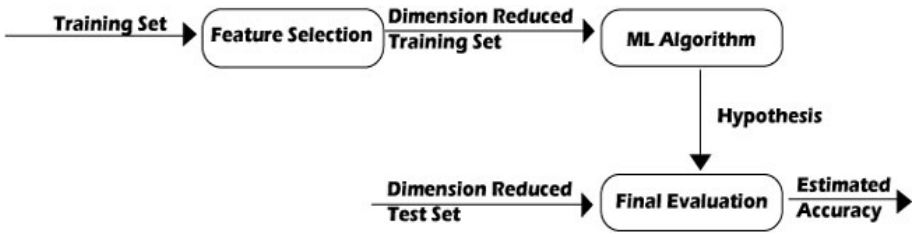


Figure 2.1: An illustration of feature subset selection.

This chapter describes feature selection techniques in detail. Section 2.1 highlights the objectives of feature selection for machine learning. Section 2.2 through Section 2.4 describes important aspects of feature selection techniques with related work.

2.1 Objectives of Feature Selection

Although feature selection adds additional computational cost to the classification task, the machine learning literature states several reasons to justify this effort:

- The number of features in the instances determine the search space that needs to be explored by the learning algorithm for a given classification task. The presence of a large number of irrelevant features unnecessarily increases the size of the search space, thus increasing the time needed for classification.
- The presence of many features, especially irrelevant and redundant ones, makes it difficult to extract knowledge such as classification rules in a way that is comprehensible to humans. Conversely, the rules based on a small number of relevant features are often concise, easier to understand and use. [41]
- The most important reason behind feature selection is that it can eliminate the effects of the curse of dimensionality.

2.2 Feature Selection Framework

Feature selection algorithms perform a search through the space of feature subsets. So, any feature selection method must address the following issues:

1. **Search strategy:** The search strategy involves organization of the search. Roughly, the search strategy can be of three types — Exhaustive, Heuristic and Random. All three search strategies are described in more detail in Section 2.4.

   Clearly, exhaustive search is impractical, as there exist 2^n possible subsets of features. A more realistic approach would be heuristic search. Forward selection and backward elimination are examples of heuristic search. Both methods usually perform $O(n^2)$ operations. In random search, the time complexity is linear to the number of iterations.

2. **Evaluation measure:** The evaluation measure determines the "goodness" of a feature or features. Most commonly used evaluation measures are information gain,

distance, dependency, consistency and accuracy. These are described in detail in Section 2.3.

3. **Stopping criterion:** A feature selection method must decide when to stop searching. One might stop adding or removing features when none of the alternatives improves upon the merit. Alternatively, one might continue to revise the feature subset as long as the merit does not degrade. Furthermore, one might continue generating feature subsets until reaching the opposite end of the search space and then select the best. [9]

2.3 Evaluation Strategy

The need for evaluation of a feature or a subset of features is common to all search strategies. Some of the common evaluation strategies are as follows.

2.3.1 Information Measures

These measures are based on ideas from information theory such as mutual information and relative entropy. An example of an information measure is the information gain.

The information gain, $Gain(C, X)$ of a given feature X, relative to the class C is the reduction in uncertainty about the value of C when we know the value of X. The uncertainty is measured in entropy. Hence,

$$Gain(C, X) = Entropy(C) - Entropy(C|X) \tag{2.1}$$

where, $Entropy(C)$ is the uncertainty about the value C and $Entropy(C|X)$ is the uncertainty about the value C when we know the value of X.

When C and X are discrete variables that take values in $\{c_1, \ ... \ , c_k\}$ and $\{x_1, \ ... \ , x_l\}$ then entropy of C is given by

$$Entropy(C) = -\sum_{i=1}^{k} P(C = c_i) log_2(P(C = c_i)) \tag{2.2}$$

The conditional entropy of C given X is

$$Entropy(C|X) = -\sum_{j=1}^{l} P(X = x_j) Entropy(C|X = x_j) \tag{2.3}$$

A feature evaluation rule based on information gain selects feature X over feature Y if $Gain(C, X) > Gain(C, Y)$. That is, the feature should be selected if it can reduce more uncertainty than the other.

Koller and Sahami [25] introduced a feature selection algorithm based on ideas from information theory. Their backward elimination algorithm uses the measure of relative entropy to minimize the information loss due to the removal of features.

2.3.2 Distance Measure

Distance measures measure the separability of the class distributions, either class conditional distributions, or the class posterior distributions. For a two-class problem, a feature X is chosen over another feature Y if X induces a greater difference between the two-class conditional distributions (or two-class posterior distributions) than Y; if the difference is zero then X and Y are indistinguishable. Some examples of distance measures are Euclidean, Kullback-Leibler and Bhattacharyya.

The RELIEF [21] algorithm can be viewed as an algorithm that uses a distance criterion by calculating the actual distance (Euclidean or other) between samples (see Subsection 2.4.3 for detail).

2.3.3 Dependence Measures

Dependence measures are based on correlation or association between features. Two features are considered correlated if knowing the value of one helps predict the value of another. In feature evaluation, correlation of a feature with the class is measured. If the correlation of feature X with class C is higher than the correlation of feature Y with class C, then feature X is preferred to Y.

Hall and Smith [17] use a correlation measure to select features. Their goal is to find

feature subsets that are highly correlated with the class but show little correlation with each other.

2.3.4 Consistency Measures

A feature evaluation rule based on consistency measures tries to find the minimum number of features that separate classes as consistently as the full set of features. An inconsistency is defined as two instances having the same feature values but different classes. Using consistency measures, both irrelevant and redundant features can be removed.

The FOCUS [3] algorithm uses a consistency measure to find an optimal feature subset (see Subsection 2.4.1 for detail).

2.3.5 Accuracy Measure

Using accuracy as a measure of feature subset goodness depends on the learning algorithm. A feature evaluation rule based on accuracy chooses the subset of features, from all possible feature subsets, that shows high prediction accuracy. The accuracy measure is used in the wrapper algorithms. This is described in Subsection 2.4.4.

2.4 Categories of Feature Selection

Based on the search strategy employed, a feature subset selection algorithm can be placed into one of three broad categories — Exhaustive search, Heuristic search and Random search.

2.4.1 Exhaustive Search

Exhaustive search exhaustively searches all possible combinations of features to find the optimal subset. Assuming there are n number of features in the original dataset, there are 2^n feature subsets to choose from. Even, if one is searching for k features, there are $\binom{n}{k}$ number of candidate subsets. It is obvious that exhaustive search always finds optimal subset. Unfortunately, the exhaustive search is impractical even for datasets with moderate numbers of features because the search space grows exponentially.

One of the famous exhaustive search algorithm is the FOCUS [3] algorithm. FOCUS was originally designed for noise-free boolean domains. FOCUS starts with an empty set and carries out breadth first search i.e. the method begins by looking at each feature in isolation, then turns to pairs of features, triples, and so forth, until it finds the minimum combination of features that predicts pure classes. This is referred to as the MIN-FEATURES bias. Following feature selection, the final feature subset is passed to an algorithm for decision-tree induction.

2.4.2 Heuristic Search

Heuristic search methods employ heuristics to avoid having to explore the full space. Since fewer subsets are examined, heuristic search is usually faster than exhaustive search. The number of subsets generated is usually $O(n^2)$. However, heuristic search is not guaranteed to return optimal subsets. But the trade off of optimality with speed is often worthwhile because of much gained speed and little loss of optimality when dealing with larger datasets.

Forward selection and backward elimination are the most common heuristic search algorithms. Forward selection starts off with an empty set of features and at each iteration, adds one feature to its current subset from the remaining features not yet selected. Forward selection chooses the feature that most increases the value of the evaluation criterion. Conversely, backward elimination starts off with a full set of features, and at each iteration, removes one feature from the full set of features that results in the least decrease in the value of the evaluation criterion. A detailed comparison of both variants, forward selection and backward elimination, can be found in [5]. They show that no method outperforms the other on all conditions in their domain and suggest that, if possible, it is preferable to test both approaches before deciding which method to use.

Caruana and Freitag [11] evaluate five greedy hillclimbing procedures on the calendar apprentice domain. They are forward selection, backward elimination, two variants of bi-directional search - forward stepwise selection and backward stepwise elimination, and backward stepwise elimination-SLASH. A decision tree construction algorithm, ID3 [35],

is used as the induction algorithm. The results indicate that bi-directional hillclimbing (the three stepwise methods) is more effective than either forward or backward selection. Feature selection was able to improve the performance of ID3 on both calendar apprentice domains. They also introduce a caching scheme to save computation time. This significantly speeds up the search but it seems to be specific to ID3.

On a similar note, John et al. [20] used both forward selection and backward elimination to minimize the cross validation error of decision tree classifiers. Kohavi [22] used hillclimbing and best-first search for feature subset selection for decision tree classifiers.

Gutlein [16], in his masters' thesis, modified forward selection to improve feature selection time. This method limits the number of available features in each step of the forward selection. This method is based on the initial ranking of the features. The features are sorted according to their merit. He proposed two different forward selection techniques which he termed as "fixed-set" technique and "fixed-width" technique. The "fixed-set" technique simply performs a forward selection on the top k features of the ranking. The "fixed-width" technique keeps the number of subset expansions at a constant level in each forward selection step. The experiments have shown that both techniques are preferable to a complete forward selection in terms of time and accuracy. The implementation of *Linear Sequential Selection* is based on the "fixed-width" technique.

2.4.3 Randomized Search

Randomized search employs randomized or probabilistic sampling processes. The motivation behind randomized search is to avoid getting stuck in local minima and to capture the interdependence of features.

The RELIEF [21] algorithm assigns a weight to each feature depending upon each features' ability in classification. The RELIEF algorithm randomly selects sample instances from the training data. The nearest instance of the same class and opposite class are found for each sample instance. An attributes' weight is determined by how well the value of the attribute distinguishes the sampled instances from the nearest hit and nearest miss. The attribute receives the highest weight if it differentiates between the different classes and

the same weight for the instance of the same class. After assigning weights, the RELIEF algorithm selects those features with weights that exceed a user-specified threshold value.

The RELIEF algorithm can handle discrete and continuous attributes but is limited to two-class problems. Kononenko [26] extended the Relief algorithm to deal with noisy, incomplete data and also, to deal with more than two classes. The extended algorithms is called Relief-F.

The LVF [30] algorithm is a Las Vegas algorithm. LVF consists of a random procedure that generates several random feature subsets and scores them according to the inconsistency they display compared to the unreduced data. Yang and Hanover [41] use a genetic algorithm to select features for neural network pattern classifiers.

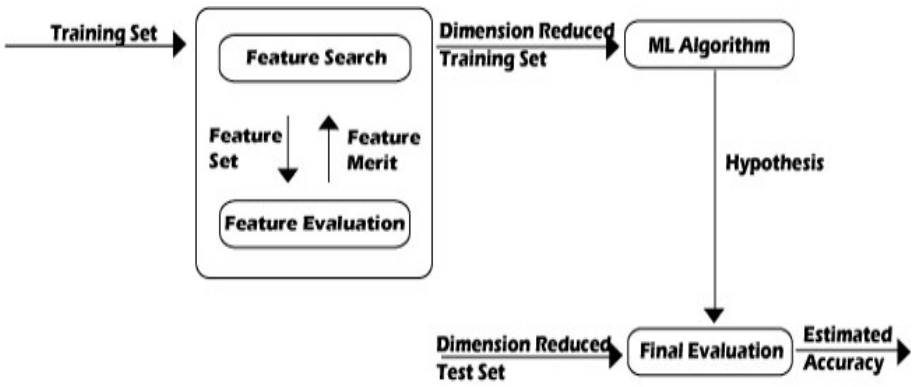
2.4.4 Filter versus Wrapper

Based on the way the learning algorithm is used, feature subset selection algorithms can be further divided into two categories — filter-based and wrapper-based. In the filter-based approach, feature selection is performed independently of the learning algorithm used for classification. In the wrapper-based approach, the feature subset selection algorithm uses the learning algorithm to evaluate each and every set of features it encounters during search process. Figure 2.2 illustrates the filter-based and wrapper-based approach.

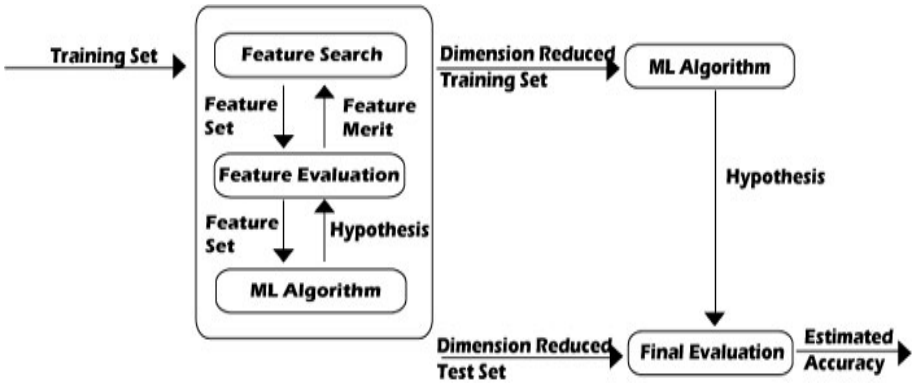
John et al. [20] were the first to introduce the wrapper-based approach for feature selection. They present two degrees of feature relevance and claim that the wrapper-based approach is able to find relevant features.

Experiments were conducted on artificial and natural domains using decision trees, ID3 and C4.5 [35], and naive Bayes as the induction algorithms. Both backward elimination and forward selection search methods were used. Results showed that the feature subset selection with the wrapper-based approach significantly improves the performance of both induction decision tree learners and naive Bayes on some datasets.

The LVW [29] algorithm is the Las Vegas algorithm that follows a wrapper-based approach. The LVW algorithm is a random procedure that generates several random feature subsets



(a) A filter-based approach



(b) A wrapper-based approach

Figure 2.2: An illustration of the filter-based and wrapper-based approach

and picks the one that has the lowest error using a decision-tree learning algorithm.

Some of the examples that use filter approaches include FOCUS [3] (see Section 2.4.1), RELIEF [21] (see Section 2.4.3) and LVF [30] (see Section 2.4.3).

The wrapper-based approaches usually carry a higher computational burden than filter-based approaches because a learning algorithm is employed to evaluate each and every set of features considered. However, the wrapper-based approaches usually perform better than the filter-based ones due to the fact that they are tuned to the specific interaction between a learning algorithm and its training set.

Chapter 3

Ensemble Learning

Ensemble approaches to classification have been the focus of much attention in recent years. In this approach, several classifiers are generated from a single classifier, the so-called base classifier, by changing the training set or the input features or the parameters of the classifier. The predictions of all base classifiers are combined into a single final prediction. The idea builds on the assumption that combining the output of multiple experts is better than the output of any single expert. Ensemble learning is illustrated in Figure 3.1.

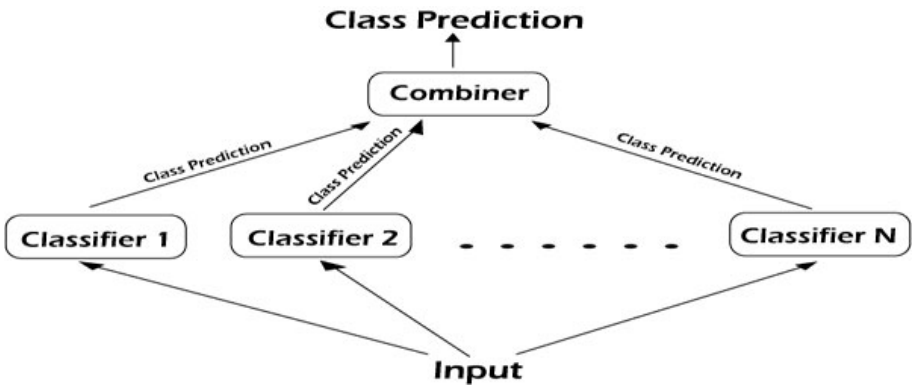


Figure 3.1: An illustration of ensemble learning.

Bagging [10] and boosting [36] are the two most popular examples of ensemble learning. Bagging creates an ensemble by training individual classifiers on bootstrap samples of the training set. Each bootstrap sample is generated by randomly selecting, with replacement,

n instances from the training set where n is the size of the training set. As a result of the sampling with replacement procedure, each classifier is trained on the average of 63.2% of the training instances. The prediction of each classifier is combined using simple voting. Each classifier votes for a particular class and the class with the majority vote on the ensemble wins.

Boosting takes a different resampling approach than bagging: sampling is proportional to an instance's weight. Initially, the instance weight is set $1/n$ for each instance. This weight is changed after each iteration based on performance. Let C_k be the current classifier where k represents classifier number, $\{1, 2, \ ... \ , m\}$. After a classifier C_k is learned, boosting changes the weights of C_k's missclassified instances by multiplying them by the factor $\beta_k = (1 - \epsilon_k)/\epsilon_k$, where ϵ_k is the sum of misclassified instance weights of the currently trained classifier C_k. The idea is to let the next classifier choose misclassified instances more often than those that are correctly classified. Finally, the predictions of all classifiers are combined by weighted voting.

Research has shown that a good ensemble should include diverse base classifiers. The diversity of the ensemble is characterized by the errors each classifier makes on different parts of the input space. In the above examples, the disturbances in the training set due to the resampling causes diverse base classifiers to be built. Another technique is to use different features for each of the base classifiers. Some of the recent work on such methods is briefly described below.

The *Random Subspace* [19] method is a simple random selection of feature subsets derived from the theory of stochastic discrimination. In this approach, one randomly selects $N' < N$ features from the N-dimensional training examples T. This procedure is repeated S times to build S number of feature subsets which are then used to construct S base classifiers. The process is very much similar to bagging, but unlike bagging, features are sampled instead of instances. Evaluation of the *Random Subspace* method with publicly available datasets has shown significant improvement in accuracy compared to those of single trees. Furthermore, it does not suffer from the curse of dimensionality like other classification methods. The *Random Subspace* method has been shown to perform better

when the dataset has a large number of features and small sample size.

Opitz [33] describes an *ensemble feature selection* technique for neural networks called *Genetic Ensemble Feature Selection* (GEFS). GEFS combines a *Random Subspace* method with a *Genetic Algorithm*. GEFS begins with creating an initial population with the *Random Subspace* method. Following this, new candidate classifiers are produced by crossover and mutation. After producing a certain number of classifiers, the process continues by selecting a new subset of candidates with a probability proportional to fitness. This process is repeated. After a predefined number of generations, the fittest individuals make up the population, which comprises the ensemble. Evaluation was done on 21 datasets from the University of Wisconsin Machine Learning repository as well as the UCI dataset repository. The results showed that GEFS compared favorably with both bagging and boosting.

Stochastic Attribute Selection Committees (SASC) [42] is an ensemble method for decision trees. The key idea behind this method is to generate different trees by stochastically varying the set of attributes available for selection at decision nodes, but keeping the distribution of the training set unchanged. C4.5 [35] is used as the base classifier. The results show that, on average, SASC is more accurate than bagging and less accurate than boosting. Zheng and Webb [43] combine SASC with Boosting to achieve accuracy higher than either Boosting or SASC alone.

Bay [6] describes a combining algorithm for nearest neighbor classifiers called *Multiple Feature Subsets* (MFS). MFS selects random subsets of features by sampling from the original set for each classifier. Each of the nearest neighbor classifiers uses the same number of features. The performance of MFS was evaluated using two different sampling methods — sampling with replacement and sampling without replacement. Twenty five datasets from UCI Machine Learning repository were used. The results showed that MFS was effective in improving accuracy.

Chapter 4
The Algorithms

This chapter describes the algorithms that have been developed for this thesis. We have implemented the algorithms for feature selection search and ensemble creation. Sections 4.1 and 4.2 describe the feature selection search algorithms in detail. Sections 4.3 and 4.4 describe the ensemble creation algorithms in detail.

4.1 Linear Sequential Selection

The motivation behind the *Linear Sequential Selection* algorithm is to find a good subset of features more efficiently than by an exhaustive feature selection algorithm. The *Linear Sequential Selection* algorithm is a heuristic wrapper-based approach that combines the rank search and forward generation schemes. The key idea of the *Linear Sequential Selection* algorithm is as follows:

> At each iteration, only investigate the best k features for potential addition to the set of features, rather than all n remaining features. If k can be held at a dataset-independent constant value without loss of quality in the resulting feature subsets, then the method is linear in the number of features.

The pseudocode for the *Linear Sequential Selection* algorithm is shown in Figure 4.1.

Since *Linear Sequential Selection* follows the *forward generation scheme*, *Linear Sequential Selection* sets $Best_{Subset}$, initially, to be empty. $Best_{Subset}$ is the best subset of features

supposed to be useful for prediction by *Linear Sequential Selection*.

```
1.    Best_Subset = null
2.    Ranked_Set = sorted attributes
3.    while true do
4.        A_k = top k attributes ∈ Ranked_Set \ Best_Subset
5.        Best_Score = 0, Best_Attribute = null
6.        for each a in A_k do
7.            if score( Best_Subset ∪ {a} ) > Best_Score then
8.                Best_Score = score( Best_Subset ∪ {a} )
9.                Best_Attribute = a
10.       if Best_Score > score( Best_Subset ) then
11.           Best_Subset = Best_Subset ∪ {Best_Attribute}
12.       else
13.           break
15.   return Best_Subset
```

Figure 4.1: The pseudocode for *Linear Sequential Selection*.

Linear Sequential Selection starts by ranking all the features of the training set. Information gain is used as default evaluation measure for ranking. Information gain is described in detail in Section 2.4.1. However, other evaluation measures can be used for ranking. All the ranked features are sorted in descending order according to their corresponding evaluation measure value and stored in $Ranked_{Set}$.

Then, *Linear Sequential Selection* operates on the top k features from the $Ranked_{Set}$, that are not in the $Best_{Subset}$ and finds the best feature for addition to the $Best_{Subset}$. *Linear Sequential Selection* is a wrapper-based approach, so the accuracy of the learning algorithm itself is used as the evaluation measure. The feature that shows highest gain in accuracy when added to the $Best_{Subset}$ is chosen as the $Best_{Attribute}$.

Finally, *Linear Sequential Selection* compares the accuracy of the $Best_{Subset}$ to the accuracy of the union of the $Best_{Subset}$ and $Best_{Attribute}$. If the addition of the $Best_{Attribute}$ to the $Best_{Subset}$ improves the accuracy, the $Best_{Attribute}$ is added to the $Best_{Subset}$.

The above procedure of selecting top k number of features, finding the $Best_{Attribute}$ and adding $Best_{Attribute}$ to the $Best_{Subset}$ is repeated while the addition of the $Best_{Attribute}$ improves the accuracy of the $Best_{Subset}$. If the addition of the $Best_{Attribute}$ does not improve the accuracy of the $Best_{Subset}$, *Linear Sequential Selection* halts.

At the end of the procedure, *Linear Sequential Selection* returns the best subset of features, i.e. $Best_{Subset}$, that is hopefully useful for analysis and future prediction.

From the above description of *Linear Sequential Selection*, we can see that the procedure of selecting the top k number of features, finding the$Best_{Attribute}$ and adding it to the $Best_{Subset}$ depends upon the number of features present in the training set, n. If, at any iteration, addition of the $Best_{Attribute}$ does not improve accuracy, then *Linear Sequential Selection* does not have to repeat the procedure any further. So *Linear Sequential Selection* can stop early without necessarily repeating the procedure n times. Furthermore, at each iteration, *Linear Sequential Selection* only investigates the top k features from the $Ranked_{Set}$, rather than all n features in the training set. This significantly increases the efficiency of *Linear Sequential Selection*. Even if *Linear Sequential Selection* has to repeat the procedure n times, the efficiency will still be better than exhaustive search or full forward selection.

4.1.1 Complexity

The time complexity of an algorithm is a measure of how much computation time is needed to run the algorithm. From the previous section, we can see that selecting a subset using *Linear Sequential Selection* involves ranking features and searching a $Ranked_{Set}$ by investigating the top k features at each iteration. The pre-processing time to rank features is of less importance, since it is done only once. Additional storage is necessary to hold the $Ranked_{Set}$. This storage requires space in the order of $O(n)$. Therefore, the major computational cost is the searching through the $Ranked_{Set}$. Since *Linear Sequential Selection* investigates the top k features at each iteration, the computational cost will be $O(kn)$ in the worst case, where n is the number features in the training set. Due to the wrapper-based characteristics of *Linear Sequential Selection*, there is an additional

cost incurred by the learning algorithm used to evaluate the worth of a feature at each addition. This will definitely increase the computational cost of *Linear Sequential Selection* compared to filter-based algorithms. However, *Linear Sequential Selection* will still be faster than other wrapper-based algorithms due to its search strategy.

4.2 Randomized Linear Sequential Selection

Linear Sequential Selection can help in getting a valid subset quickly on average, but it can not guarantee to find the optimal feature subset. This is because *Linear Sequential Selection* adopts a hill climbing heuristic. *Linear Sequential Selection* selects the best feature sequentially hoping that an absolute minimal (optimal) subset will emerge. This will surely speed up the selection process, but if the some locally minimal is found, *Linear Sequential Selection* cannot backtrack and undo past additions.

---

1. $Best_{Subset} = null$
2. $Ranked_{Set}$ = sorted attributes
3. **while** *true* **do**
4. A_k = top k attributes $\in$ Ranked$_{Set} \setminus$ Best$_{Subset}$
5. **replace least favorable** m **attributes in** A_k **with randomly selected attributes from** $Ranked_{Set} \setminus$ (Best$_{Subset}$ $\cup$ A$_k$)
6. $Best_{Score} = 0$, $Best_{Attribute} = null$
7. **for each** a **in** A_k **do**
8. **if** $score(\ Best_{Subset} \cup \{a\}\ ) > Best_{Score}$ **then**
9. $Best_{Score} = score(\ Best_{Subset} \cup \{a\}\ )$
10. $Best_{Attribute} = a$
11. **if** $Best_{Score} > score(\ Best_{Subset}\ )$ **then**
12. $Best_{Subset} = Best_{Subset} \cup \{Best_{Attribute}\}$
13. **else**
14. **break**
16. **return** $Best_{Subset}$

Figure 4.2: The pseudocode for *Randomized Linear Sequential Selection*.

The general procedure for *Randomized Linear Sequential Selection* is similar to *Linear Sequential Selection*. The only difference is the incorporation of randomization in the search space. The pseudocode of the *Randomized Linear Sequential Selection* is shown in

Figure 4.2.

Randomized Linear Sequential Selection incorporates randomization in the search space as follows: In each iteration, after the top k features are selected, *Randomized Linear Sequential Selection* replaces the least favorable features in the top k feature set by randomly selecting features from the $Ranked_{Set}$ that are not already selected. Any number of features m in the top k feature set can be replaced, so long as $m \leq k$. If m number of features have to be replaced, then *Randomized Linear Sequential Selection* selects m number of features from the $Ranked_{Set}$ randomly. If all the features in the top k feature set are replaced, i.e. $m = k$, *Randomized Linear Sequential Selection* algorithm performs completely random selection process. The remainder of the procedure is the same as for *Linear Sequential Selection*.

The use of randomization in selecting features is merely a convenient way to explore the other possibilities. *Randomized Linear Sequential Selection* considers features usually not considered by *Linear Sequential Selection*. Above all, *Randomized Linear Sequential Selection* tries to avoid getting trapped in a local optimum by allowing a degree of randomness to enter the subset generation process.

4.2.1 Complexity

The time complexity of *Randomized Linear Sequential Selection* is similar to *Linear Sequential Selection*. Although there is additional cost in the randomization, i.e. number of features in the top k feature set that are replaced by randomly selected features from the $Ranked_{Set}$, but this cost is negligible. Therefore, the time complexity will still be $O(kn)$.

4.3 Feature Selection Ensemble

Most of the known feature ensemble creation algorithms use the random subspace method [6, 19, 33, 42, 43]. In the random subspace method, each individual classifier uses its own subset of the given features for both training and testing. These subsets contain elements randomly chosen out of the whole feature set, where the size of the feature subset is usually the same for each classifier in the ensemble. The main goal of these approaches is not to

select good subsets of features, but to create diverse classifiers.

In this thesis, we propose a new ensemble creation algorithm called *Feature Selection Ensemble*, which follows an approach different to most of the known ensemble creation algorithms. The *Feature Selection Ensemble* algorithm does not select subsets of feature randomly. Instead, it applies an algorithm that selects good subsets of features for the generation of the classifiers in the ensemble. In principle, any known algorithm for feature selection can be used. The only modification that is needed is to prevent the feature selection algorithm from returning the same subset of features multiple times. This is achieved by removing all selected features from the training set, such that subsequent creation of a new classifier does not have access to the previously selected features. The pseudocode of the *Feature Selection Ensemble* is shown in Figure 4.3.

```
1.    T := the number of classifiers,
2.    H := a committee consisting of T classifiers.
3.    for each t from 1 to T
4.        select features, F, from a training set, D.
5.        H_t = build classifier with only selected features, F.
6.        D = remove selected features, F, from D, i.e. D \ F.
7.    return H
```

Figure 4.3: The pseudocode for *Feature Selection Ensemble*.

The *Feature Selection Ensemble* algorithm starts with a full set of features in the training data. To begin with, the algorithm selects features from the training set using a feature selection algorithm. Only the selected features are used for generating an individual classifier. After the classifier is built, the selected features are removed from the training set. This ensures that for the generation of subsequent classifiers, the training set does not contain any previously selected features. This procedure, i.e. selecting features, building a classifier and removing selected feature, is repeated until either the desired number of classifiers are built, or all the features in the training set are exhausted. All the individual classifiers form the ensemble in the *Feature Selection Ensemble*.

At the classification stage, for a given test example, *Feature Selection Ensemble* combines the prediction of each individual classifier by voting. We use a voting method that uses the probabilistic predictions produced by each classifier in the ensemble. With this method, each classifier returns a distribution over all classes. All individual distributions are summed up into one final distribution. The class with the highest score (highest probability) wins the vote, and serves as the predicted class of *Feature Selection Ensemble* for the given test example.

With the *Feature Selection Ensemble* algorithm, each individual classifier in the ensemble is built from a different subset of features. The feature subsets are mutually exclusive. As mentioned earlier, a good ensemble consists of classifiers that are as diverse as possible. Building each individual classifier in the ensemble with a different subset of features, as in *Feature Selection Ensemble*, ensures such diversity.

4.4 Random Feature Ensemble

In the *Feature Selection Ensemble* algorithm, any feature selection algorithm can be used for classifier ensemble generation. In this section, we describe a new ensemble creation algorithm called *Random Feature Ensemble* which exploits the behavior of the *Randomized Linear Sequential Selection* algorithm for classifier ensemble generation. We believe that the randomization introduced in the *Randomized Linear Sequential Selection* algorithm will force the generation of more diverse classifiers for the ensemble. The *Randomized Linear Sequential Selection* algorithm is described in detail in Section 4.2.

The *Random Feature Ensemble* algorithm is different from the *Feature Selection Ensemble* algorithm in the sense that the *Random Feature Ensemble* algorithm does not remove features from the training set. During the building process, a full set of features is provided each time and then the *Randomized Linear Sequential Selection* algorithm is applied in order to select features for the classifier. This process is repeated until the desired number of classifiers are built. The pseudocode of the *Random Feature Ensemble* is shown in Figure 4.4.

```
1.    T := the number of classifiers,
2.    H := a committee consisting of T classifiers.
3.    for each t from 1 to T
4.        select features, F , from a training set, D, using Randomized Linear
          Sequential Selection.
5.        H_t = build classifier with only selected features, F.
6.    return H
```

Figure 4.4: The pseudocode for *Random Feature Ensemble*.

Finally, the results of all individual classifiers are combined by voting as in *Feature Selection Ensemble* (See Section 4.3 for detail).

4.5 Chapter Summary

This chapter presents four new algorithms — two, *Linear Sequential Selection* and *Randomized Linear Sequential Selection*, are feature selection search methods and the other two, *Feature Selection Ensemble* and *Random Feature Ensemble*, are ensemble creation methods.

Linear Sequential Selection examines only the top k features at each iteration of the selection process for possible addition. This makes *Linear Sequential Selection* faster than exhaustive search as well as most other heuristic feature selection algorithms. Random Linear Sequential Selection is developed to improve classification accuracy of *Linear Sequential Selection*. The idea is to explore other features, not considered during regular sequential selection.

Feature Selection Ensemble and *Random Feature Ensemble* are ensemble creation techniques based on feature subsets. For *Feature Selection Ensemble*, any known feature selection algorithm can be used. After each iteration, the selected features are removed from the training set to prevent the feature selection algorithm from returning the same subset of features again. *Random Feature Ensemble* explicitly incorporates *Random Linear Sequential Selection* to select more diverse sets of feature.

Chapter 5
Evaluation Methodology

In order to evaluate the practical value of the algorithms — *Linear Sequential selection*, *Randomized Linear Sequential Selection*, *Feature Selection Ensemble* and *Random Feature Ensemble* — we have implemented the algorithms in the WEKA framework [13]; and ran a number of experiments. The performance of the algorithms are evaluated. This chapter reviews the general methodology used to evaluate the performance of the algorithms. Section 5.1 describes metrics used to evaluate performance of the algorithms. Section 5.2 looks at datasets used and their properties in detail. Section 5.3 and Section 5.4 describe the cross-validation method and the information gain measure respectively. Section 5.5 reviews three machine learning algorithms — naive Bayes, J48 and IBk. Finally, Section 5.6 describes greedy feature selection algorithm used for comparison.

5.1 Metrics

The performance is an aspect of behavior that can be measured quantitatively (i.e. by some value). The following measures are used to examine the performance of the algorithms.

5.1.1 Classification Accuracy

Classification accuracy is a commonly used measure to assess the performance of the algorithm. Classification accuracy is defined as the percentage of test examples correctly classified by the classifier. The classifier is better if classification accuracy is higher. This

is because the classifier tests well on unseen data that is not involved in inducing the classifier.

In this thesis, when there is a significant difference, it means that the difference is statistically significant at the 5%level according to a paired two-sided t-test, each pair of data points consisting of the estimates obtained in ten 10-fold cross-validation run for the two learning schemes being compared.

5.1.2 Speed of feature selection methods

This measure is used to assess the time required to select features. Since the feature selection algorithm is used as a preprocessor, the feature selection algorithm adds extra cost, i.e. time complexity, in the classification task of the classifier. The feature selection algorithm is better if the feature selection algorithm consumes less time to select features.

In this thesis, time corresponds to the sum of feature selection and classification time.

5.1.3 Number of features selected

This is a measure for assessing the size of data. The feature selection algorithm is better if the feature selection algorithm selects a smaller number of features. The small number of features leads to faster learning, fewer potential hypothesis and simpler end results.

5.2 Datasets

The experiments are performed on 20 datasets from UCI Machine Learning repository [8] and 4 publicly available microarray datasets [4, 2, 34, 15]. The datasets and their properties are listed in Table 5.1. Some datasets only contain nominal features or numeric features, and others contain a mix of nominal features and numeric features. The number of features ranges from 15 to 7129. Some datasets contain missing values and others do not. About half of the datasets represent two-class domains and the other half represent multi-class domains. Thus, these datasets give good representation of real-world machine learning problems.

Dataset	Features	Numeric	Nominal	Missing	Instances	Classes
credit-a	15	6	9	yes	690	2
labor	16	8	8	yes	57	2
vote	16	0	16	yes	435	2
primary-tumor	17	0	17	yes	330	21
lymphography	18	3	15	no	148	4
vehicle	18	18	0	no	846	4
hepatitis	19	6	13	yes	155	12
segment	19	19	0	no	2390	7
credit-g	20	7	13	no	1000	2
colic	21	7	15	yes	368	2
autos	25	15	10	yes	205	6
colic.ORIG	27	7	20	yes	368	2
ionosphere	34	34	0	no	351	2
soyabean	35	0	35	yes	683	19
kr-vs-kp	36	0	36	no	3196	2
anneal	38	6	32	no	898	5
anneal.ORIG	38	6	32	yes	898	5
sonar	60	60	0	no	208	2
audiology	69	0	69	yes	226	24
arrhythemia	279	206	73	yes	452	13
colon-cancer	2000	2000	0	no	62	2
lymphoma	4026	4026	0	yes	45	2
CNS	7129	7129	0	no	60	2
leukemia	7129	7129	0	no	34	2

Table 5.1: The 20 UCI and 4 microarray datasets, and their properties.

The four microarray datasets used in the experiments are as follows:

Colon-Cancer contains 62 samples with 2000 features. The samples are collected from colon-cancer patients. Among the 62 samples, 40 tumor biopsies are from tumors (class: negative) and 22 normal (class: positive) biopsies are from healthy parts of the colons of the same patients. [4]

Lymphoma is a microarray dataset containing 47 samples, 24 of them are from "germinal centre B-like" group while 23 are "activated B-like" group. Each sample has 4026 features. [2]

CNS stands for Central Nervous System. This microarray dataset contains 60 samples with 7129 genes (or features). The two classes, Class0 and Class1, describe the outcome of the treatment of a Central Nervous System Embryonal Tumor. Class0 represents survival and Class1 represents failure. [34]

Leukemia is a microarray dataset containing 72 bone marrow samples. We merged the training and test set together (34 and 38 samples). The two classes describe two types of acute leukemia, 47xALL and 25xAML. Each sample has 7129 attributes. [15]

5.3 Cross-validation

Cross-validation is a standard procedure for evaluating learning methods. In k-fold cross-validation, the training data is randomly divided into k mutually exclusive subsets of approximately equal size. A learning algorithm is tested k times; each time one set is used as testing data while remaining others, i.e. $k-1$, sets are used as training data. The average of all k test accuracies is the final estimation of the k-fold cross-validation. The random sampling of the data may result in unrepresentative training data. Stratification is often applied during k-fold cross-validation. Stratification ensures that each class is properly represented in both training and testing sets. In practice, 10-fold stratified cross-validation is repeated 10 times in order to provide a stable estimate. A detailed comparison of popular evaluation methods is described in [39].

5.4 Information Gain

The information gain measure is used to rank features from training set (For detail see Chapter 2). *InfoGainAttribEval* is the WEKA [13] implementation of the information gain evaluation measure.

5.5 Learning Algorithms

Three machine algorithms are used as a basis for comparing the effects of feature selection with no feature selection — naive Bayes, J48 and IB1. Each learning algorithm represents a different approach of learning.

5.5.1 Naive Bayes

The naive Bayes algorithm is based on Bayes' rule of conditional probability. The naive Bayes algorithm computes conditional probabilities of the classes given the feature values present in the instance and picks the class with highest posterior probability. The feature values are assumed to be statistically independent within each class. Equation 5.1 shows the naive Bayes formula.

$$p(C_i|f_1, f_2, \dots, f_n) = \frac{p(C_i) \prod_{j=1}^{n} p(f_j|C_i)}{p(f_1, f_2, \dots, f_n)} \tag{5.1}$$

The left side of Equation 1.1 is the posterior probability of class C_i given the feature values, ($f_1, f_2, \dots, f_n$), observed in the test instance. Since the denominator of the right side fraction does not depend on C_i and the values of the features f_n are given, the denominator of the right side fraction is constant. Therefore, learning with the naive Bayes classifier involves simply estimating the probabilities in the right side fraction of Equation 5.1, i.e. the numerator, from the training instances. The result is a probabilistic summary for each of the possible classes.

5.5.2 J48

J48 is the implementation of the C4.5 [35] decision tree algorithm in the WEKA framework [13]. J48 builds decision tree from training instances by using an information theoretic measure. J48 examines the information gain, i.e. change in entropy, that results from choosing a feature that splits the training instances into subsets corresponding to the values of the feature. If the entropy of the class labels in these subsets is less than the entropy of the class labels in the full training set, then information has been gained. The feature with highest information gain is the one used to make a node of the tree. The algorithm then recurses on the subsets to form subtrees, terminating when a given subset contains instances of only one class.

5.5.3 IB1

IB1 is the implementation of instance-based learner, also called nearest neighbor. Instance-based learners are lazy learners because learning is delayed until classification time. In instance-based learners, each new instance is compared with existing ones using Euclidean distance metric and the closest existing instance is used to assign the class to the new one [39]. Equation 5.2 shows the distance metric employed by instance-based learner.

$$D(x, y) = \sqrt{\sum_{j=1}^{n} (x_j - y_j)^2} \tag{5.2}$$

Equation 5.2 gives the distance between two instances x and y; x_j and y_j refer to the jth feature value of instance x and y respectively.

IBk is an extension to IB1, called k nearest neighbor. IBk assigns the most common class from the k nearest neighbor to the test instance where k is a parameter set by the user. Instance-based learning is explained in more detail in [1].

5.6 Greedy Algorithm

The *GreedyStepwise* algorithm is the WEKA [13] implementation of a greedy hill climbing approach for feature selection. The *GreedyStepwise* adds the best feature (or deletes the

worst feature) at each iteration. The best feature is the one when added to the current set yields high merit under a certain evaluation criteria. Conversely, the worst feature is the one when removed from the current set yields high merit under certain evaluation criteria. The *GreedyStepwise* algorithm is used to compare results with *Linear Sequential Selection*

5.7 Chapter Summary

This chapter explains the methods used to evaluate the performance of the 4 algorithms. For *Linear Sequential Selection*, the speed and number of features selected are the primary evaluation criteria. The *Linear Sequential Selection* is considered successful if the speed and number of features selected are faster and small respectively without compromising the classification accuracy of the learning algorithm. The classification accuracy is the primary evaluation criteria for the other 3 algorithms — *Randomized Linear Sequential Selection*, *Feature Selection Ensemble* and *Random Feature Ensemble*.

Chapter 6

Experimental Results

This chapter analyzes the experimental results obtained from the evaluation (as described in chapter 5) of the 4 algorithms: *Linear Sequential Selection*, *Randomized Linear Sequential Selection*, *Feature Selection Ensemble* and *Random Feature Ensemble*. This chapter is divided into 4 sections. Each section focusses on one algorithm. Section 6.1 analyzes results of *Linear Sequential Selection* with 3 ML algorithms – naive Bayes, J48 and IBk, in terms of accuracy, time and number of features. Section 6.2, Section 6.3 and Section 6.4 analyze the accuracy of *Randomized Linear Sequential Selection*, *Feature Selection Ensemble* and *Random Feature Ensemble* with 3 ML algorithms respectively.

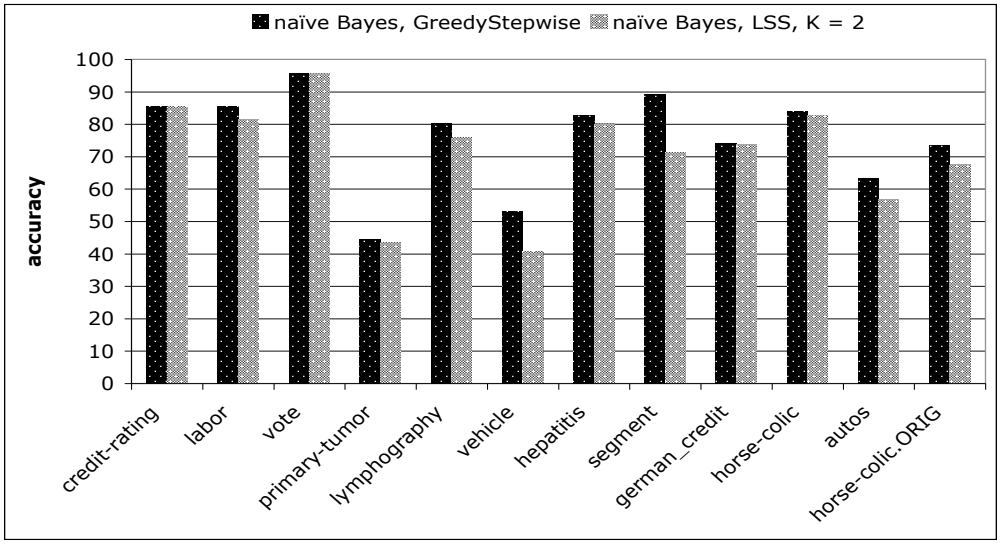
6.1 Results – Linear Sequential Selection

This section shows the performance of *Linear Sequential Selection* using 3 ML algorithms. Three different values of k (2, 5 and 10) are used. The obtained results are analyzed by comparing them with the respective ML algorithm using GreedyStepwise and without feature selection in terms of accuracy, number of attributes and time.

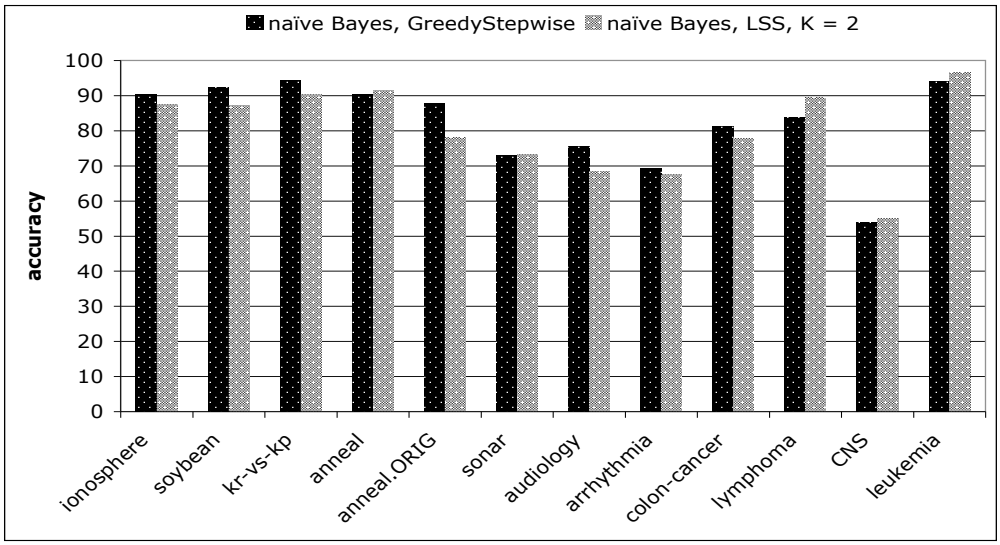
6.1.1 Accuracy

Classification accuracy of naive Bayes

Figure 6.1 compares the classification accuracy of the naive Bayes using the features obtained by GreedyStepwise and default settings of *Linear Sequential Selection* (i.e. $k =$

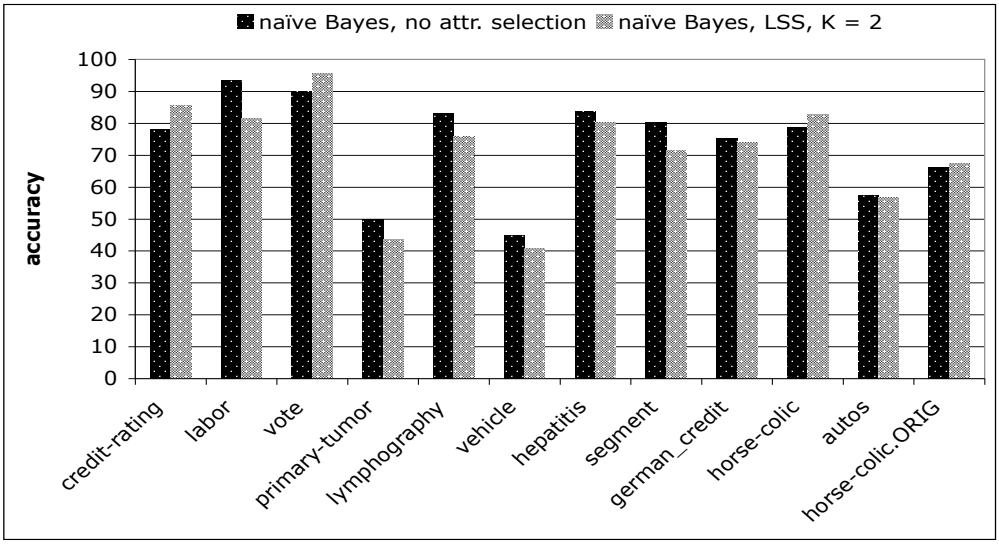


(a)

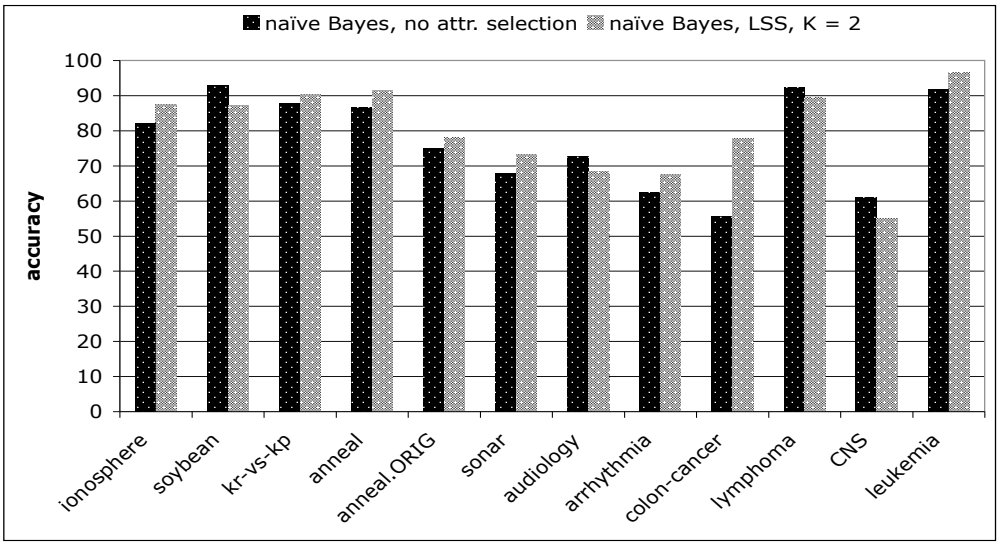


(b)

Figure 6.1: The classification accuracy of naive Bayes with GreedyStepwise and default settings of *Linear Sequential Selection* (i.e. $k = 2$).

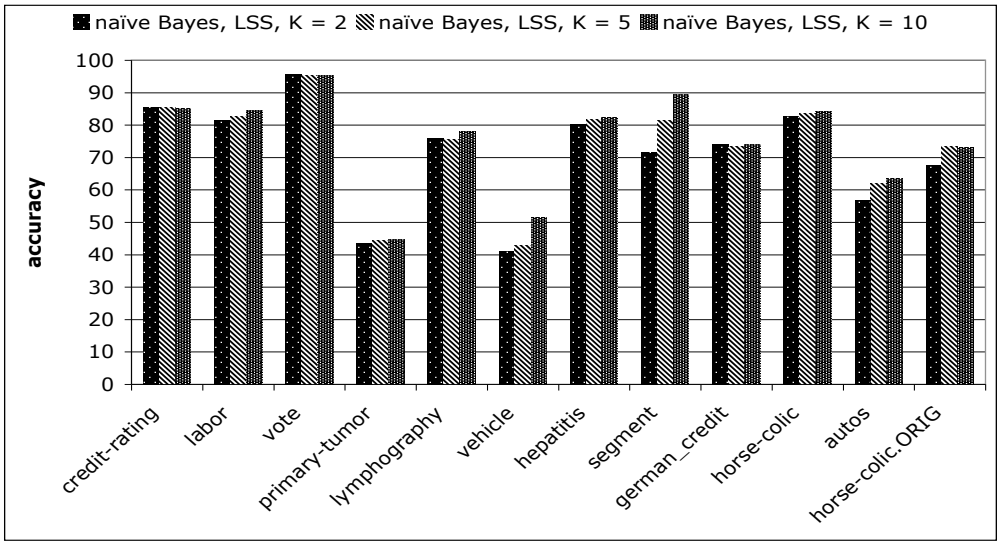


(a)

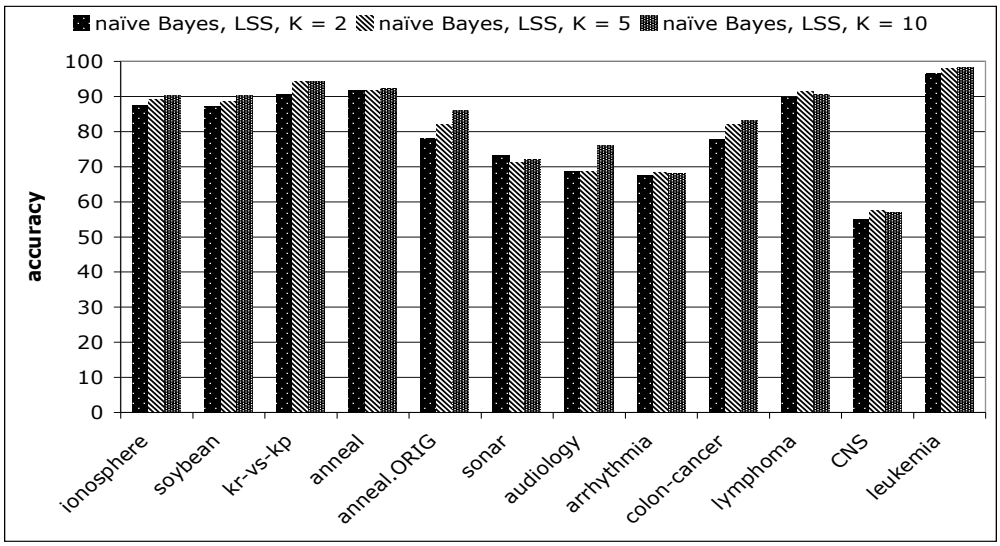


(b)

Figure 6.2: The classification accuracy of naive Bayes without feature selection and with default settings of *Linear Sequential Selection* (i.e. $k = 2$).



(a)



(b)

Figure 6.3: The classification accuracy of naive Bayes using *Linear Sequential Selection* with $k = 2$, 5 and 10.

2) on 24 datasets. The figure shows that *Linear Sequential Selection* using naive Bayes is, on average, less accurate. Seventeen datasets show degradation in the classification accuracy which is significant for 7 datasets (anneal.ORIG, audiology, horse-colic.ORIG, kr-vs-kp, segment, soyabean and vehicle). The remaining 7 datasets maintain or show slight improvements in accuracy.

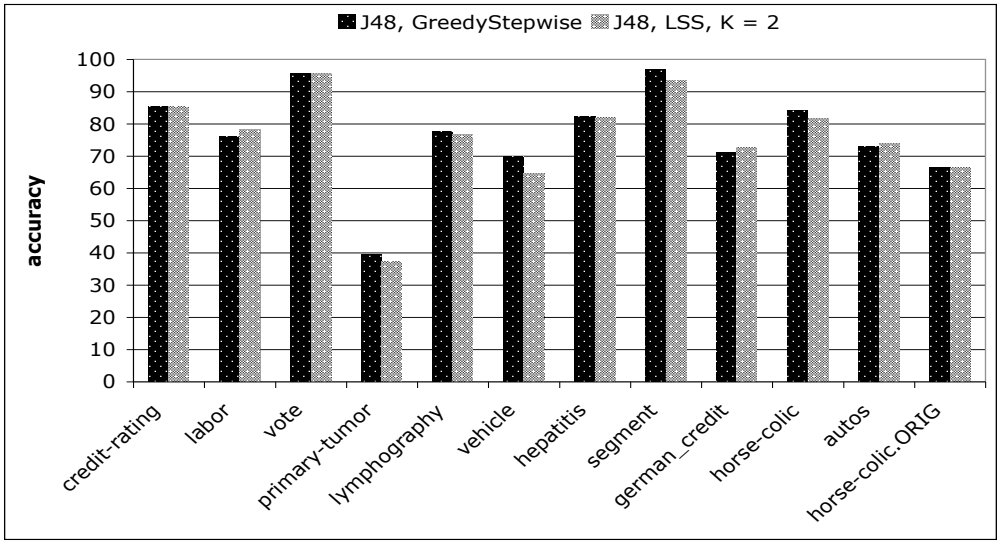
Figure 6.2 compares the classification accuracy of naive Bayes without feature selection and using the attributes obtained by default settings of *Linear Sequential Selection* (i.e. $k = 2$) on 24 datasets. The figure shows that naive Bayes using *Linear Sequential Selection* shows improvement in the classification accuracy on 12 of 24 datasets, 6 (anneal, audiology, credit-rating, ionosphere, kr-vs-kp and vote) them are significant. The remaining 12 datasets show degradation in the classification accuracy, 6 (labor, lymphography, primary-tumor, segment, soybean and vehicle) of them are significant.

Figure 6.3 compares classification accuracy of naive Bayes using different settings of k in *Linear Sequential Selection* on 24 datasets. The values of k in *Linear Sequential Selection* are set to $k = 2$, $k = 5$ and $k = 10$. The classification accuracy, on average, is improved with increase in the value of k. This can be seen in 16 out of 24 datasets. Three datasets (credit-rating, vote and sonar) show decrease in the accuracy. For two datasets (lymphography and german_credit), the accuracy decreases when k is 5 but increases when k is 10. For 3 datasets (horse-colic.ORIG, arrhythmia and lymphoma), the accuracy increases when k is 5 and then decreases when k is 10.

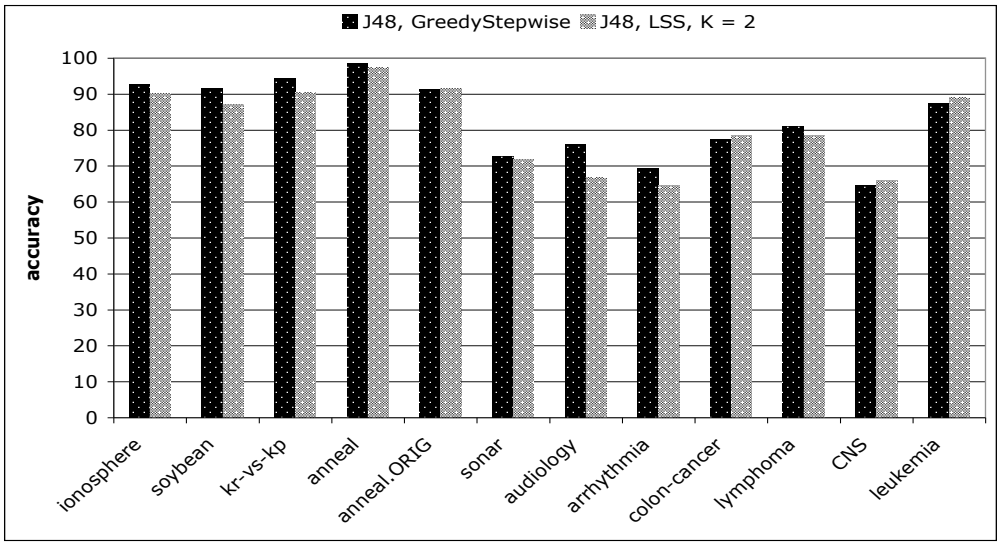
Classification accuracy of J48

Figure 6.4 compares the classification accuracy of J48 using the features obtained by GreedyStepwise and default settings of *Linear Sequential Selection* (i.e. $k = 2$) on 24 datasets. The figure shows that J48 using *Linear Sequential Selection* is less accurate on 14 of 24 datasets which is significant for 8 datasets (anneal, arrhythmia, audiology, horse-colic, kr-vs-kp, segment, soybean and vehicle). The other remaining 10 datasets maintain or show improvement in the classification accuracy.

Figure 6.5 compares the classification accuracy of J48 without feature selection and using

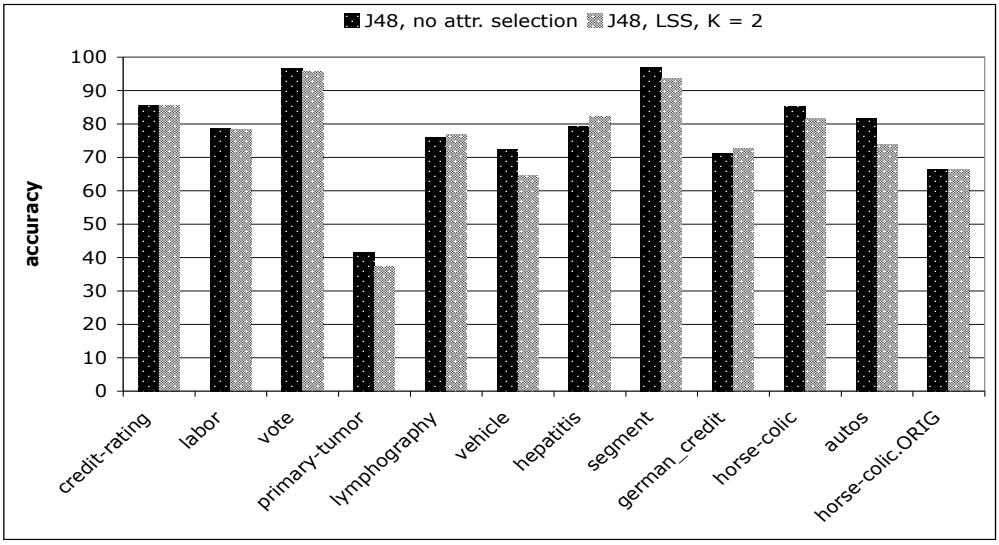


(a)

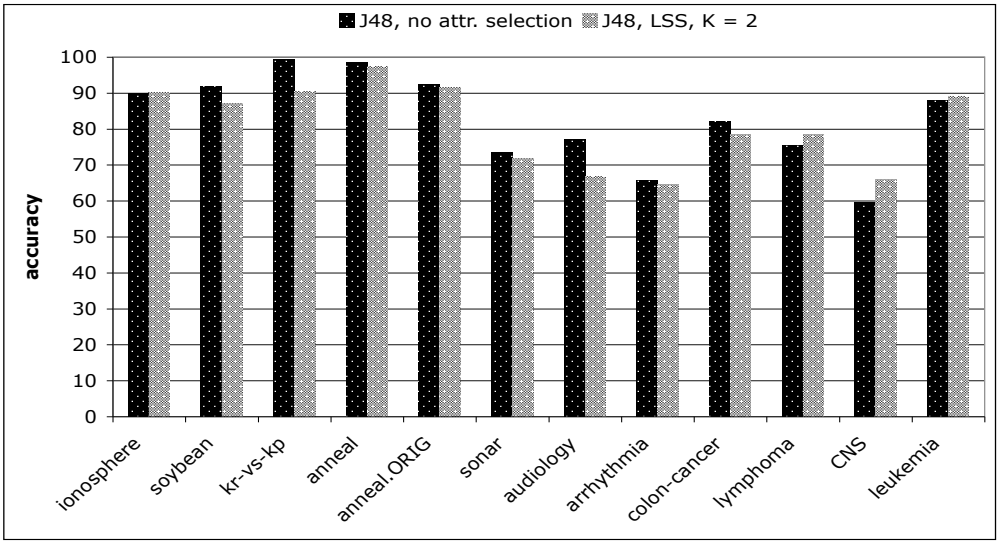


(b)

Figure 6.4: The classification accuracy of J48 with GreedyStepwise and default settings of *Linear Sequential Selection* (i.e. $k = 2$).

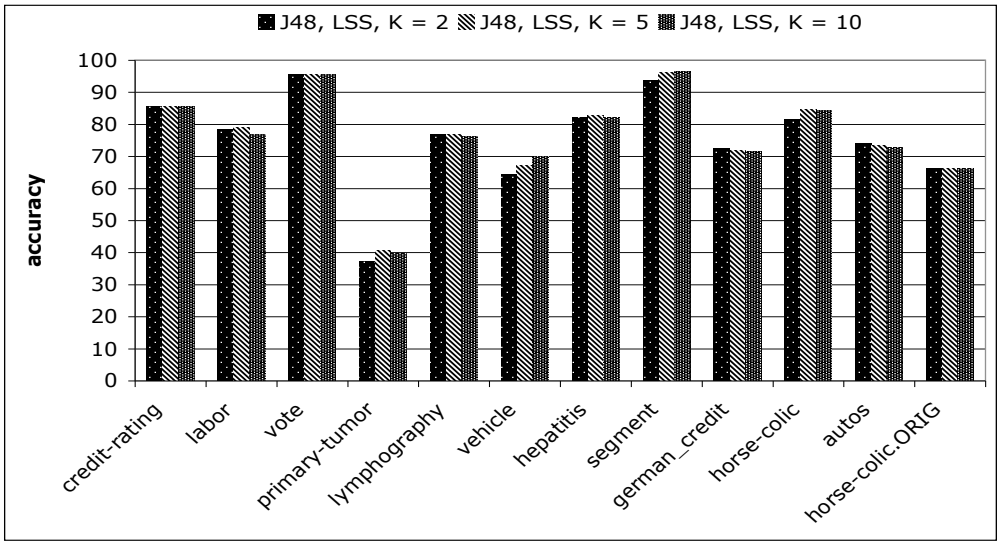


(a)

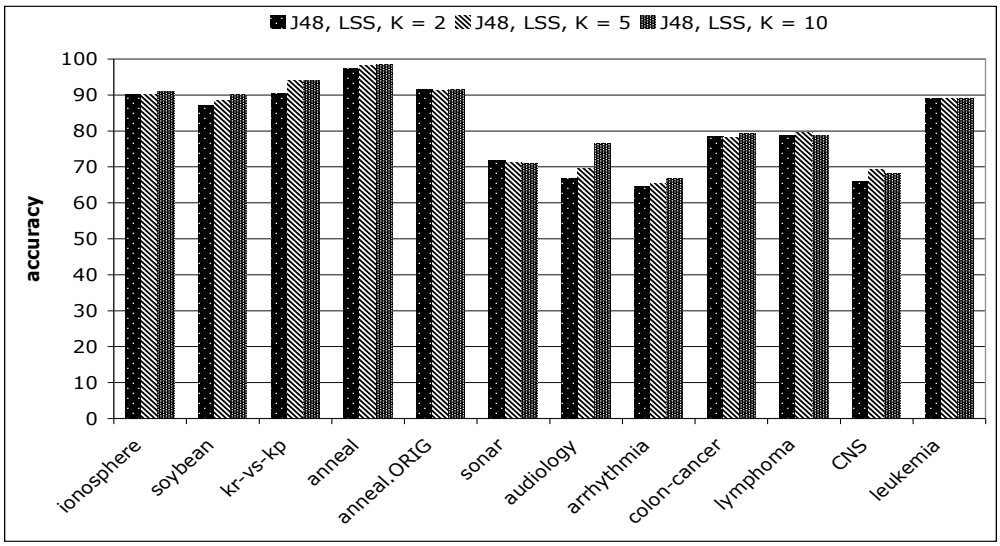


(b)

Figure 6.5: The classification accuracy of J48 without feature selection and with default settings of *Linear Sequential Selection* (i.e. $k = 2$).



(a)



(b)

Figure 6.6: The classification accuracy of J48 using *Linear Sequential Selection* with $k =$ 2, 5 and 10.

the attributes obtained by default settings of *Linear Sequential Selection* (i.e. $k = 2$) on 24 datasets. The figure shows that J48 using *Linear Sequential Selection* is less accurate on 16 of 24 datasets, 8 (anneal, audiology, autos, horse-colic, kr-vs-kp, segment, soybean and vehicle) of them are significant. The remaining 8 datasets maintain or show improvement in the classification accuracy.

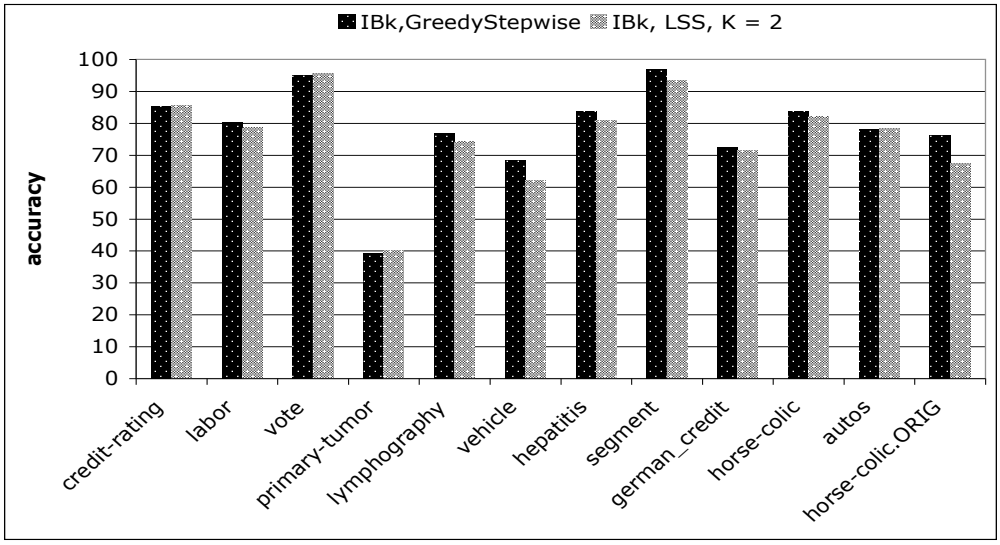
Figure 6.6 compares classification accuracy of J48 using different settings of k in *Linear Sequential Selection* on 24 datasets. The values of k in *Linear Sequential Selection* are set to $k = 2$, $k = 5$ and $k = 10$. On 10 datasets, the classification accuracy is maintained or improved with increase in the value of k. Four datasets (credit-rating, german_credit, autos and sonar) show decrease in the accuracy. For 3 datasets (ionosphere, anneal.ORIG, and colon-cancer), the accuracy decreases when k is 5 and then increases when k is 10. For 7 datasets (labor, primary-tumor, lymphography, hepatitis, horse-colic, lymphoma and CNS), the accuracy increases when k is 5 but decreases when k is 10.

Classification accuracy of IBk

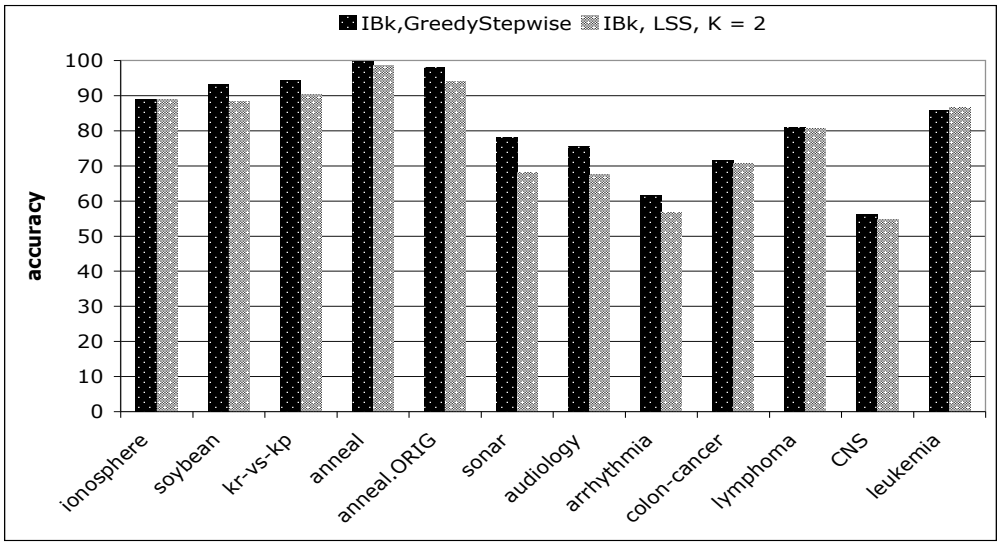
Figure 6.7 compares the classification accuracy of IBk using the features obtained by GreedyStepwise and default settings of *Linear Sequential Selection* (i.e. $k = 2$) on 24 datasets. The figure shows that IBk using *Linear Sequential Selection* is less accurate on 17 of 24 datasets which is significant for 10 datasets (lymphography, vehicle, hepatitis, segment, horse-colic.ORIG, soybean, kr-vs-kp, anneal.ORIG, sonar, audiology and arrhythmia). The other remaining 7 datasets maintain or show improvement in the classification accuracy.

Figure 6.8 compares the classification accuracy of IBk without feature selection and using the attributes obtained by default settings of *Linear Sequential Selection* (i.e. $k = 2$) on 24 datasets. The figure shows that IBk using *Linear Sequential Selection* is less accurate on 14 of 24 datasets. The remaining 10 datasets maintain or show improvement in the classification accuracy.

Figure 6.9 compares classification accuracy of IBk using different settings of k in *Linear Sequential Selection* on 24 datasets. The values of k in *Linear Sequential Selection* are set

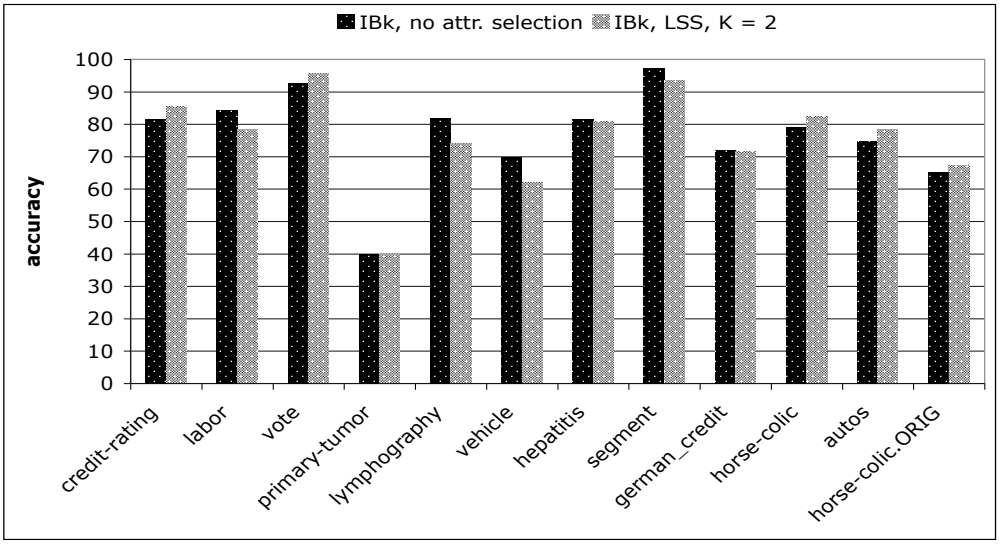


(a)

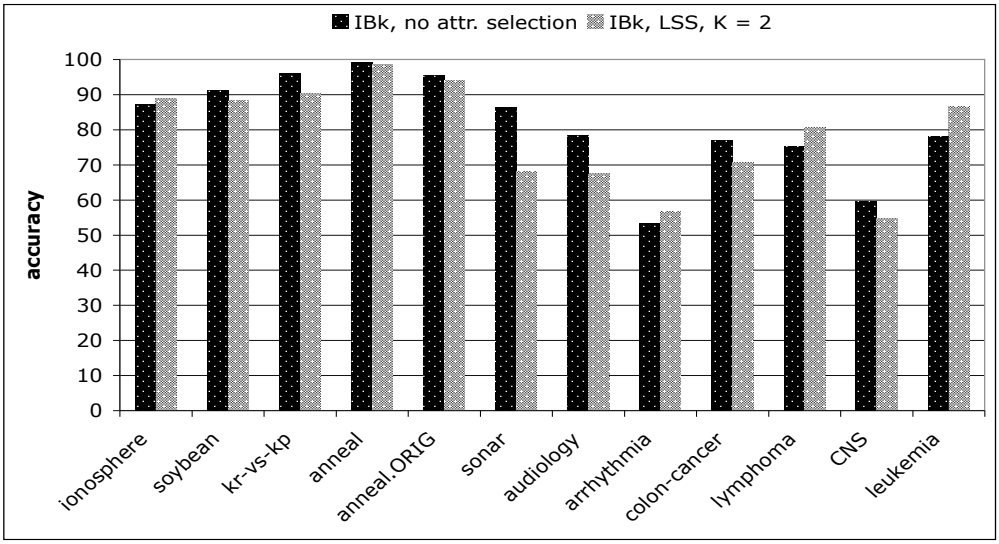


(b)

Figure 6.7: The classification accuracy of IBk with GreedyStepwise and default settings of *Linear Sequential Selection* (i.e. $k = 2$).

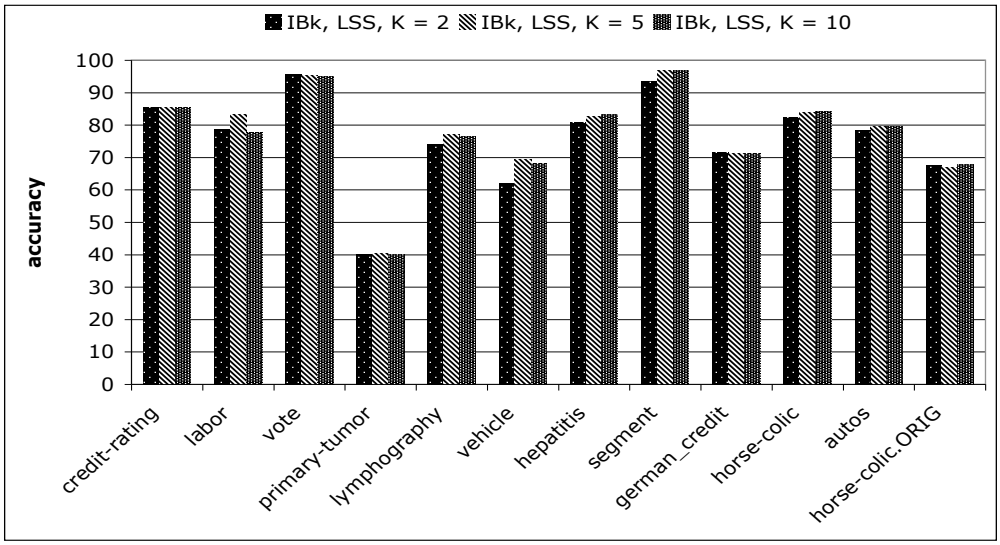


(a)

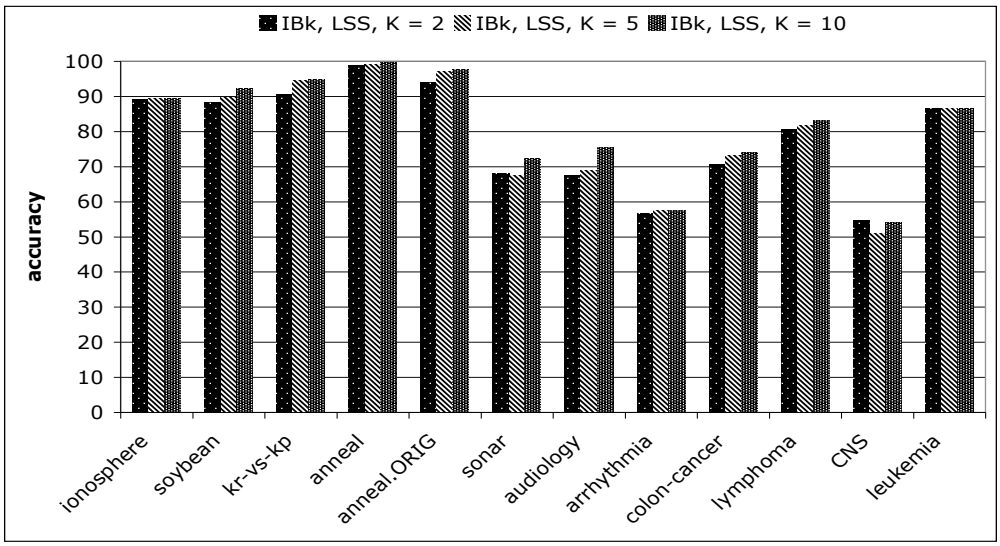


(b)

Figure 6.8: The classification accuracy of IBk without feature selection and with default settings of *Linear Sequential Selection* (i.e. $k = 2$).



(a)



(b)

Figure 6.9: The classification accuracy of IBk using *Linear Sequential Selection* with $k =$ 2, 5 and 10.

to $k = 2$, $k = 5$ and $k = 10$. For 14 datasets, the classification accuracy is maintained or improved with increase in the value of k. Three datasets (credit-rating, vote and german_credit) show decrease in the accuracy. For 2 datasets (horse-colic.ORIG and CNS), the accuracy decreases when k is 5 and then increases when k is 10. For 5 datasets (labor, primary-tumor, lymphography, vehicle, and arrhythmia), the accuracy increases when k is 5 but decreases when k is 10.

6.1.2 Number of Attributes

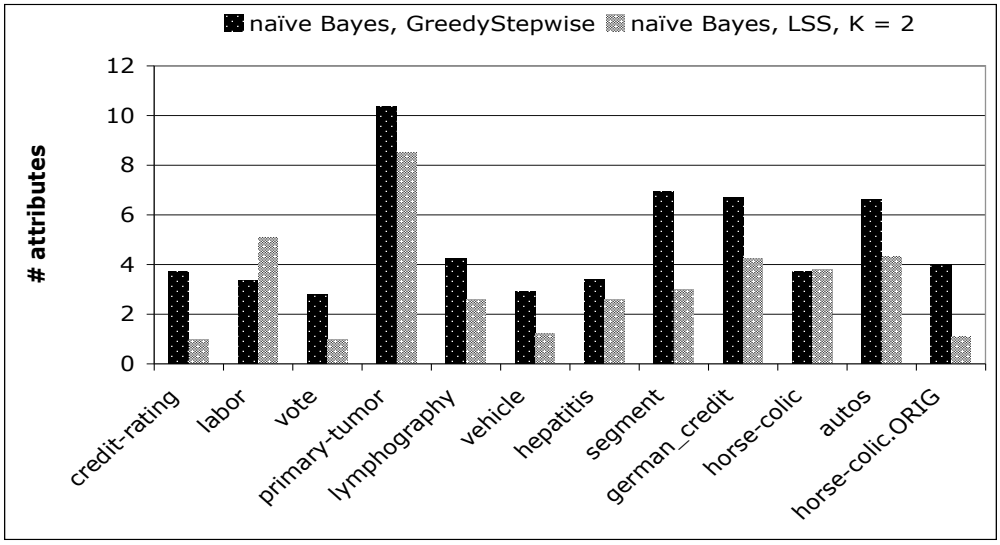
Number of attributes using naive Bayes

Figure 6.10 compares the number of features selected by GreedyStepwise and default setting (i.e. $k = 2$) of *Linear Sequential Selection* using naive Bayes on 24 datasets. The *Linear Sequential Selection* selects fewer features than GreedyStepwise on 21 of 24 datasets. On 3 datasets (labor, horse-colic.ORIG and leukemia), *Linear Sequential Selection* selects more features.

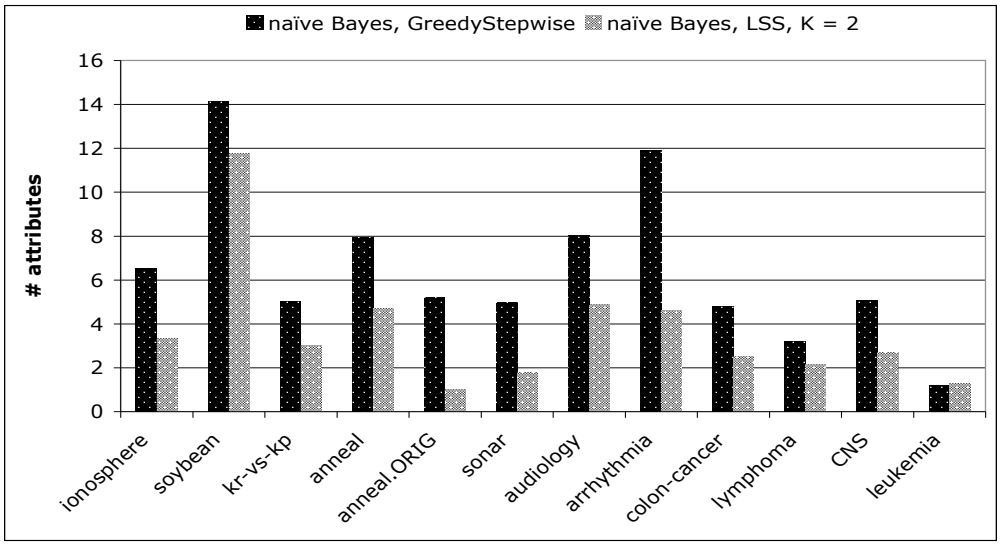
Figure 6.11 compares the number of features selected by different settings of k in *Linear Sequential Selection* using naive Bayes. The values of k in *Linear Sequential Selection* are set to $k = 2$, $k = 5$ and $k = 10$. The figure shows that the number of features selected by *Linear Sequential Selection* increases with increase in the value of k on 17 of 24 datasets. One dataset (labor) shows decrease in the number. On 5 datasets (hepatitis, horse-colic, anneal, colon-cancer, and leukemia), the number of selected features decreases when k is 5 and then increases when k is 10. For 1 dataset (soyabean), the number of selected features increases when k is 5 but decreases when k is 10.

Number of attributes using J48

Figure 6.12 compares the number of features selected by GreedyStepwise and default setting (i.e. $k = 2$) of *Linear Sequential Selection* using J48 on 24 datasets. The *Linear Sequential Selection* selects fewer features than GreedyStepwise on 20 of 24 datasets. On 3 datasets (segment, german_credit and horse-colic.ORIG), the *Linear Sequential Selection* selects more features. *Linear Sequential Selection* and GreedyStepwise select same number of features on 1 dataset (vote).

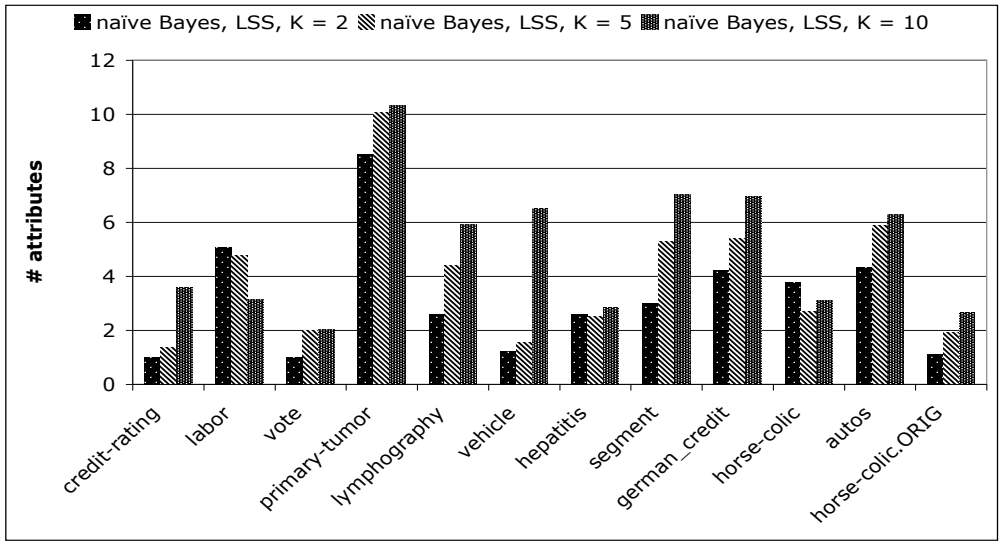


(a)

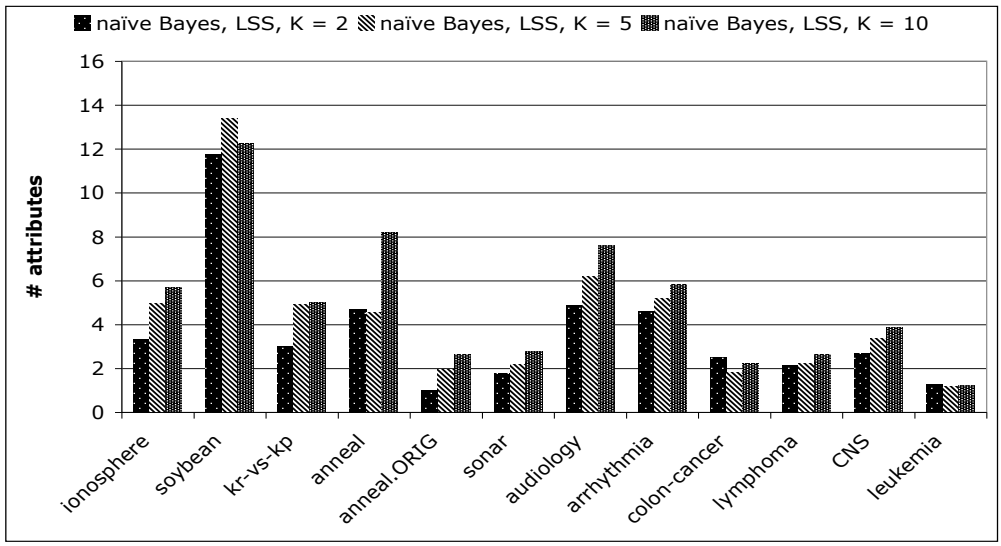


(b)

Figure 6.10: The number of attributes selected by GreedyStepwise and default settings of *Linear Sequential Selection* (i.e. $k = 2$) using naive Bayes.

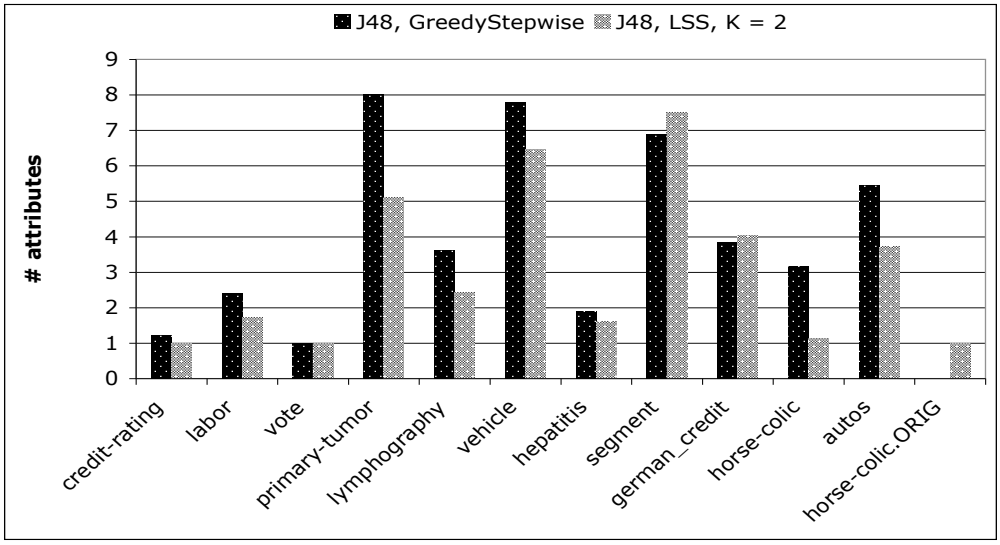


(a)

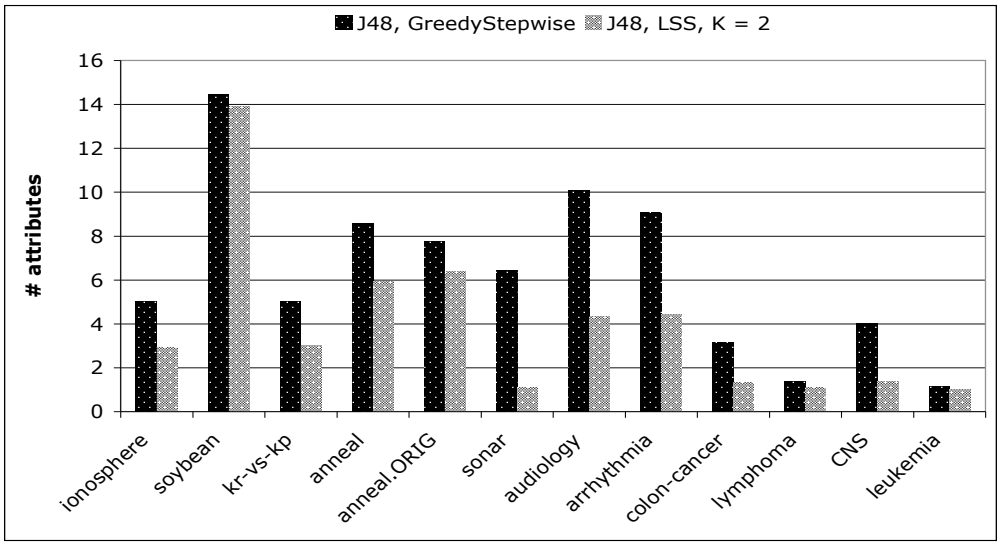


(b)

Figure 6.11: The number of attributes selected by *Linear Sequential Selection* with settings $k = 2$, 5 and 10 using naive Bayes.

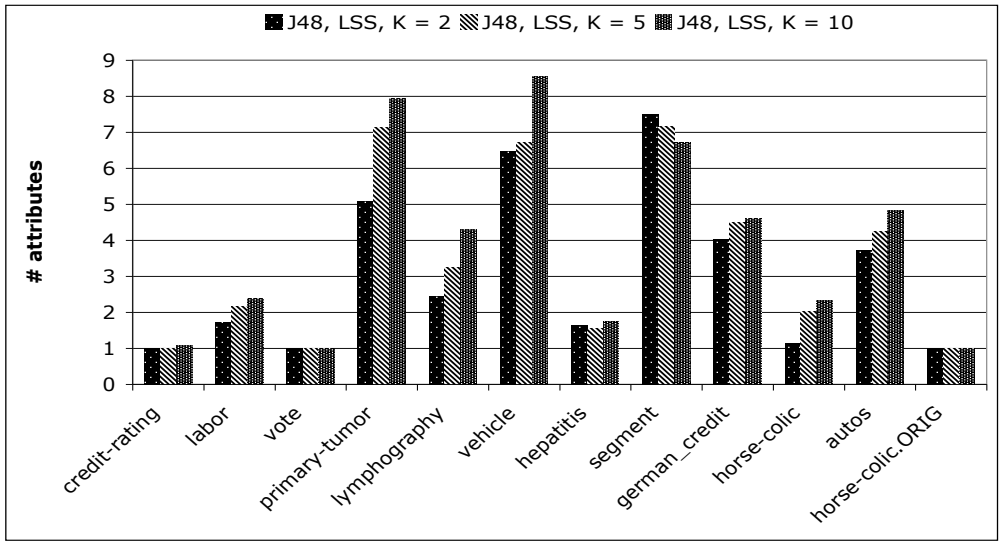


(a)

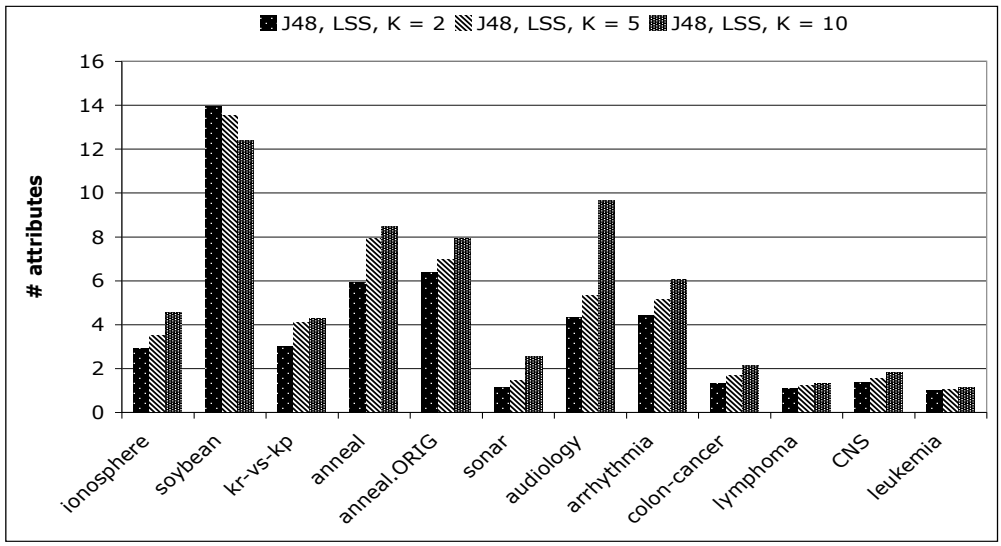


(b)

Figure 6.12: The number of attributes selected by GreedyStepwise and default settings of *Linear Sequential Selection* (i.e. $k = 2$) using J48.

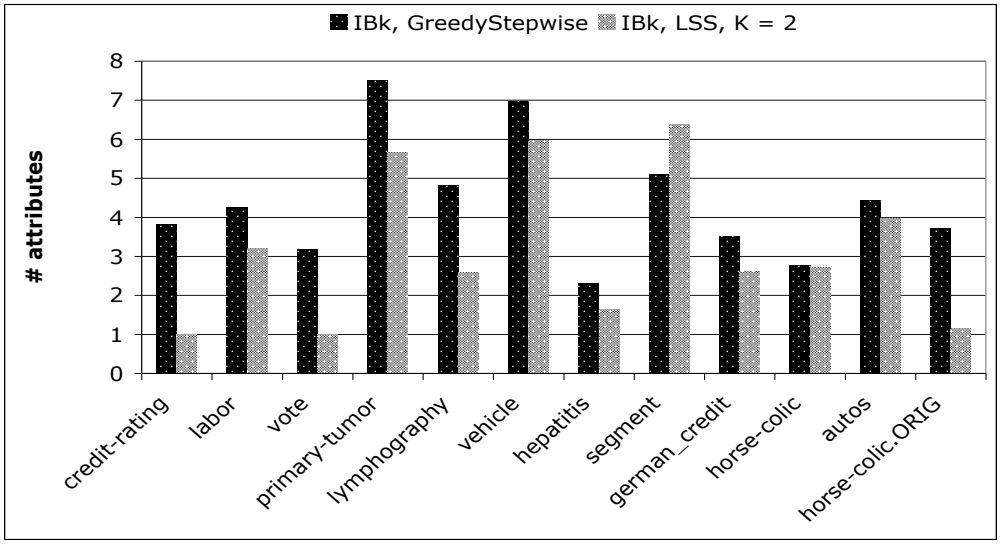


(a)

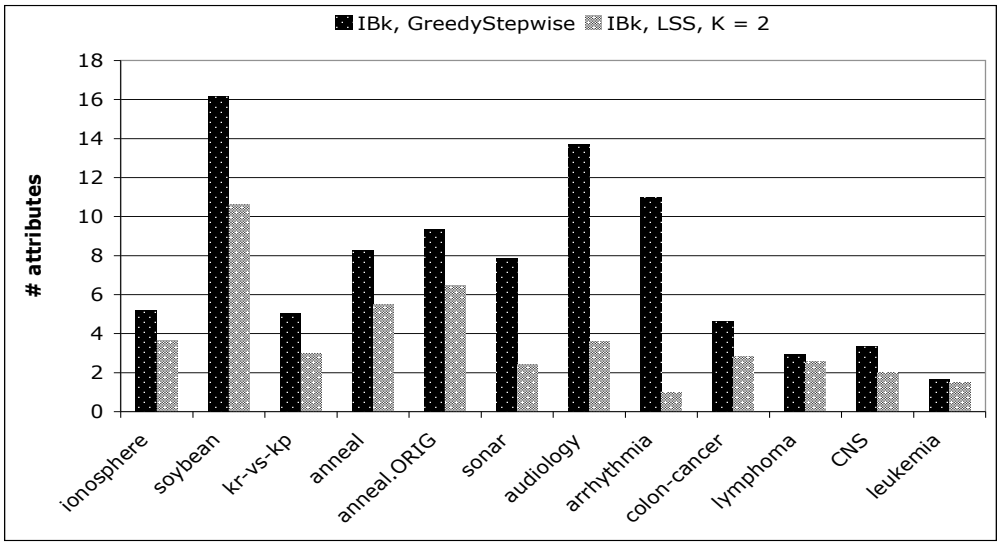


(b)

Figure 6.13: The number of attributes selected by *Linear Sequential Selection* with settings $k = 2$, 5 and 10 using J48.

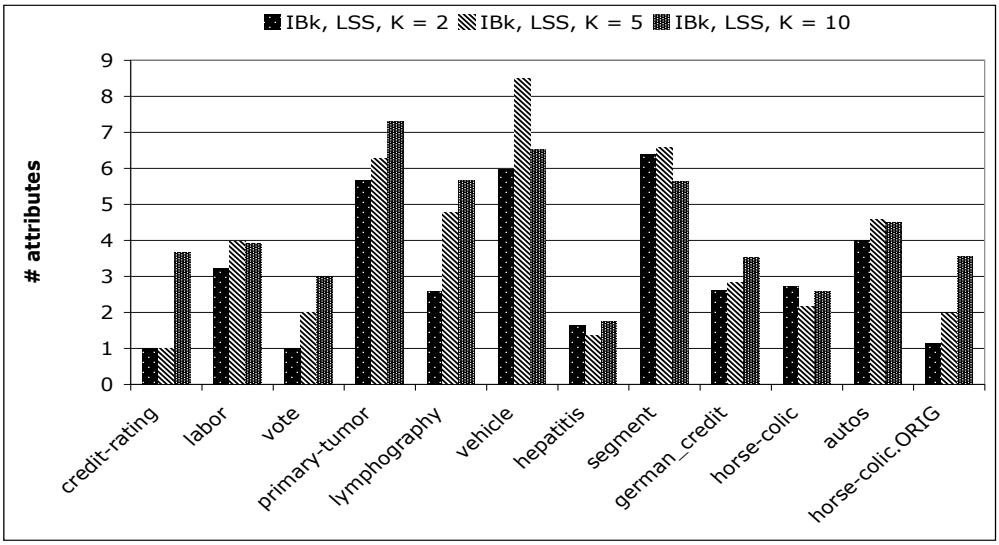


(a)

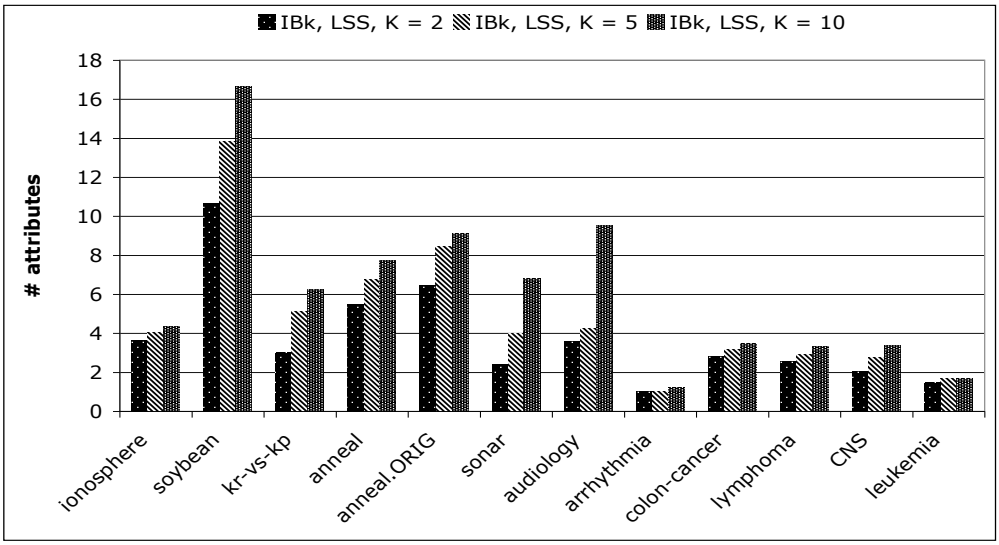


(b)

Figure 6.14: The number of attributes selected by GreedyStepwise and default settings of *Linear Sequential Selection* (i.e. $k = 2$) using IBk.



(a)



(b)

Figure 6.15: The number of attributes selected by *Linear Sequential Selection* with settings $k = 2$, 5 and 10 using IBk.

Figure 6.13 compares the number of features selected by different settings of k *Linear Sequential Selection* using J48 on 24 datasets. The values of k in *Linear Sequential Selection* are set to $k = 2$, $k = 5$ and $k = 10$. The figure shows that the number of features selected by *Linear Sequential Selection* increases with increase in the value of k on 19 of 24 datasets. Two datasets (segment and soyabean) show decrease in the number. For 1 dataset (hepatitis), the number of selected features decreases when k is 5 and then increases when k is 10. For 2 datasets (vote and horse-colic.ORIG), the number of selected features remains same.

Number of attributes using IBk

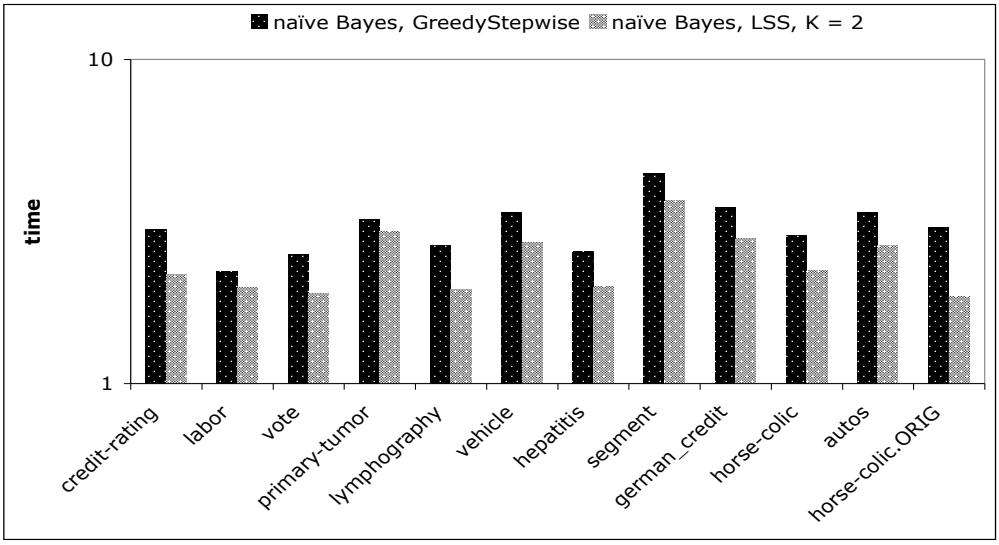
Figure 6.14 compares the number of features selected by GreedyStepwise and default setting (i.e. $k = 2$) of *Linear Sequential Selection* using IBk on 24 datasets. The *Linear Sequential Selection* selects fewer features than GreedyStepwise on 23 of 24 datasets. Only on 1 dataset (segment), the *Linear Sequential Selection* selects more features.

Figure 6.15 compares the number of features selected by different settings of k *Linear Sequential Selection* using IBk on 24 datasets. The values of k in *Linear Sequential Selection* are set to $k = 2$, $k = 5$ and $k = 10$. The figure shows that the number of features selected by *Linear Sequential Selection* increases with increase in the value of k on 19 of 24 datasets. For 2 datasets (hepatitis and horse-colic), the number of selected features decreases when k is 5 and then increases when k is 10. For 3 datasets (labor, vehicle and autos), the number of selected features increases when k is 5 and then decreases when k is 10.

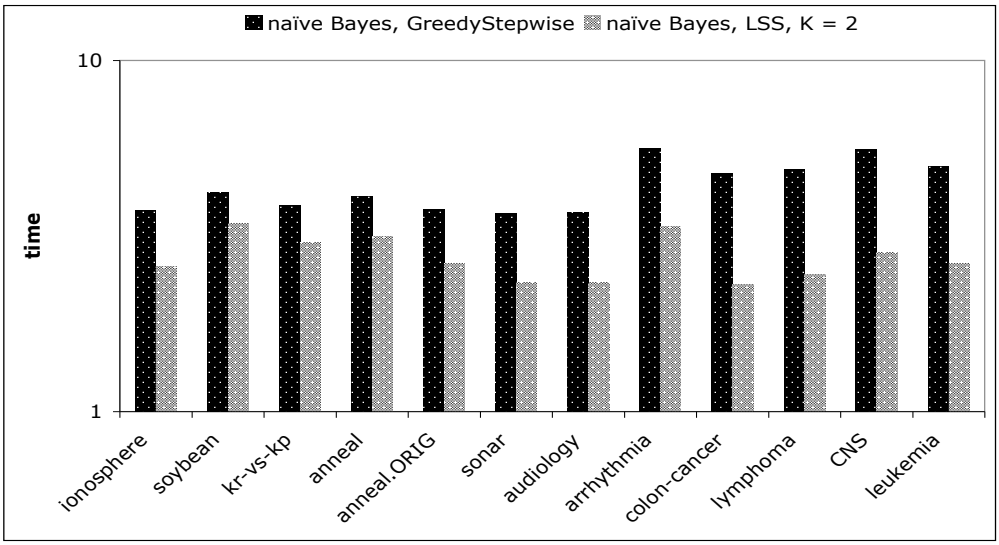
6.1.3 Time

Time using naive Bayes

Figure 6.16 shows the runtime of naive Bayes using GreedyStepwise and *Linear Sequential Selection* with default setting (i.e. $k = 2$). Note that the runtime is the sum of feature selection time and classification time and it is expressed on a logarithmic scale. The figure shows that the runtime of naive Bayes using *Linear Sequential Selection* is faster than naive Bayes using GreedyStepwise on all 24 datasets.

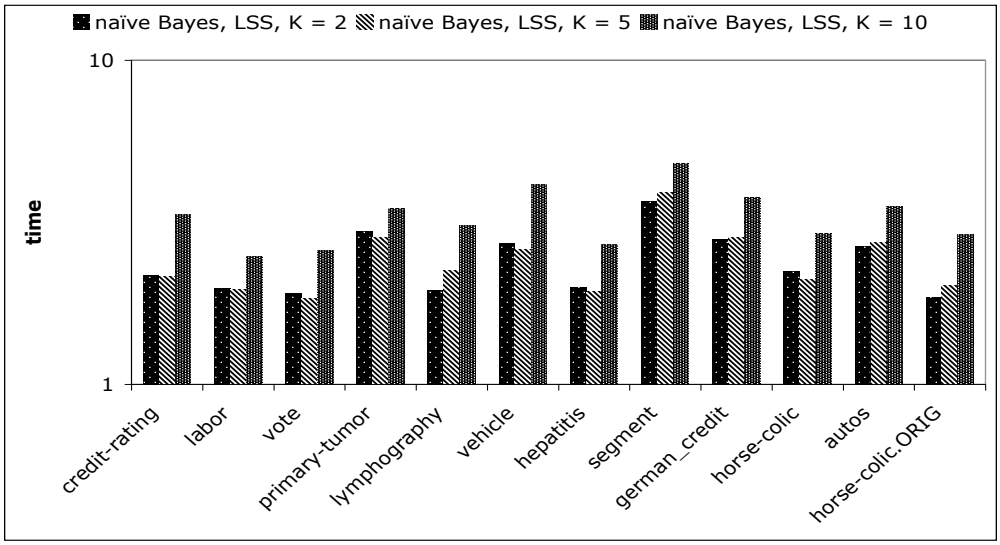


(a)

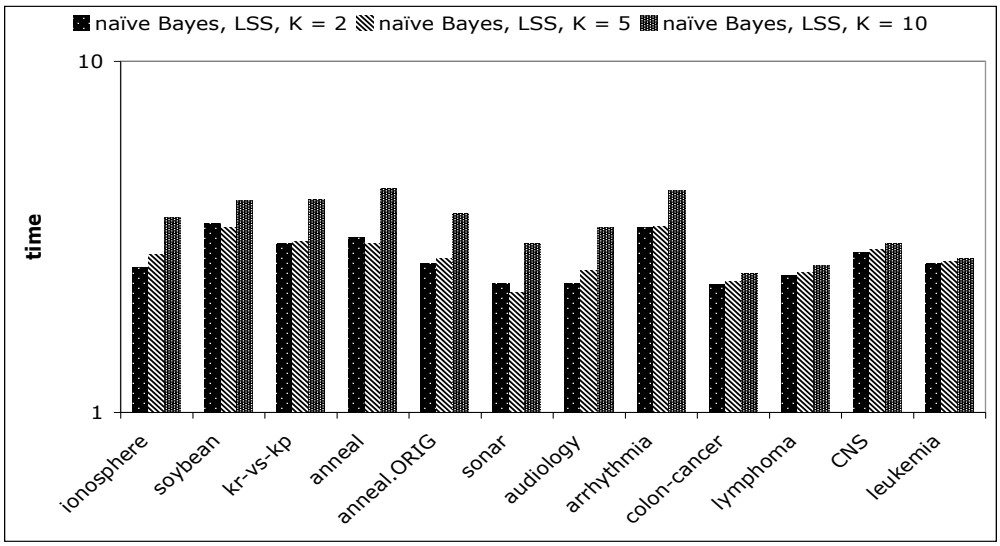


(b)

Figure 6.16: The runtime of GreedyStepwise and *Linear Sequential Selection* with default setting (i.e. $k = 2$) using naive Bayes.

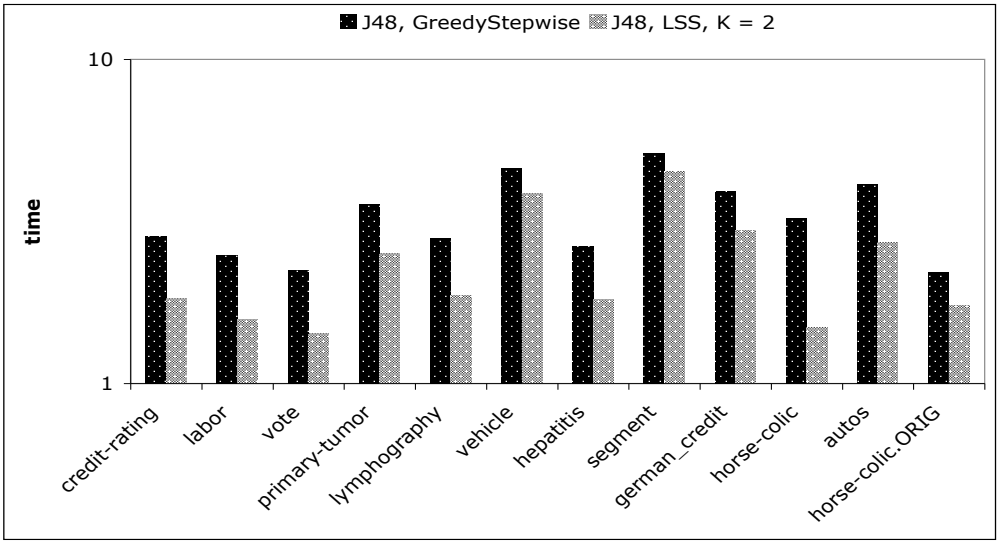


(a)

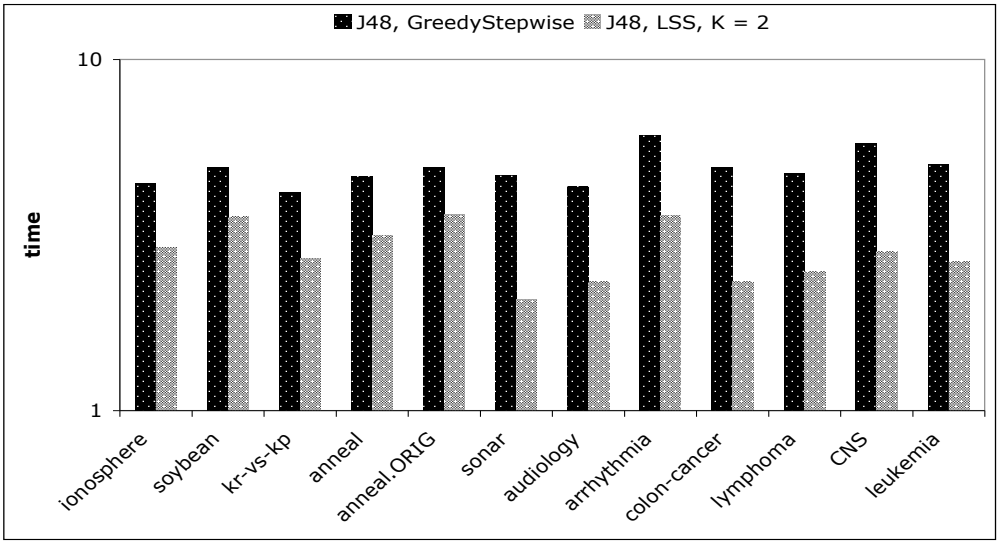


(b)

Figure 6.17: The runtime of emphLinear Sequential Selection with settings $k = 2$, 5 and 10 using naive Bayes.

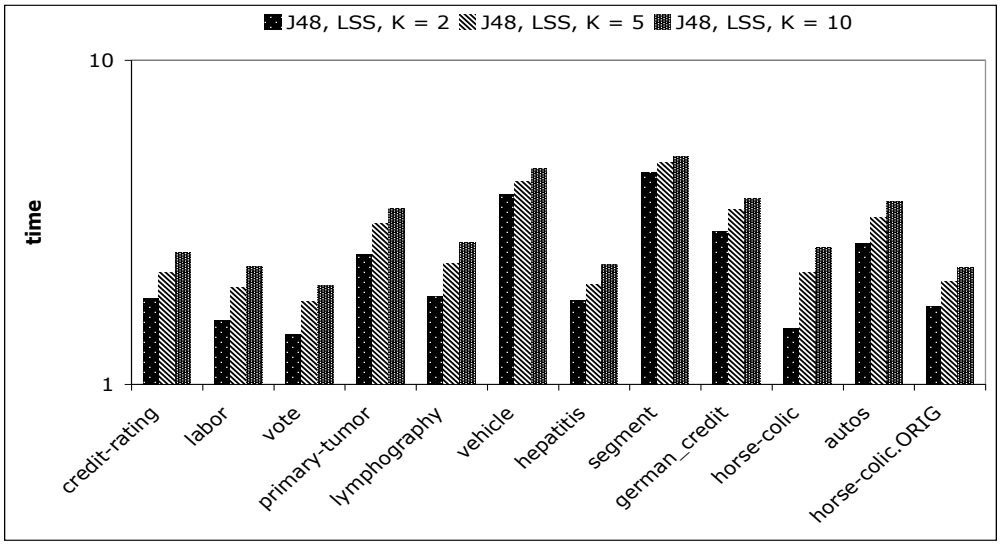


(a)

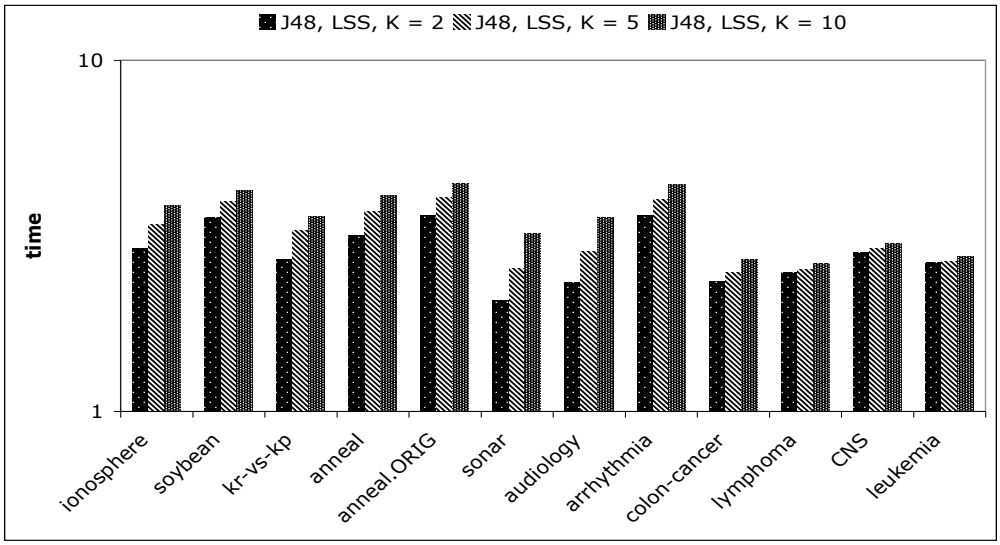


(b)

Figure 6.18: The runtime of GreedyStepwise and *Linear Sequential Selection* with default setting (i.e. $k = 2$) using J48.

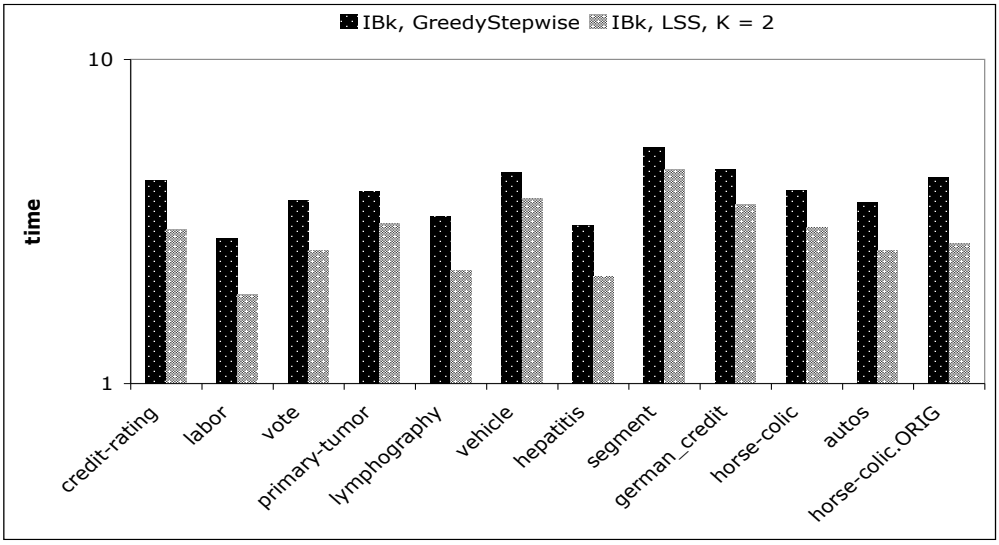


(a)

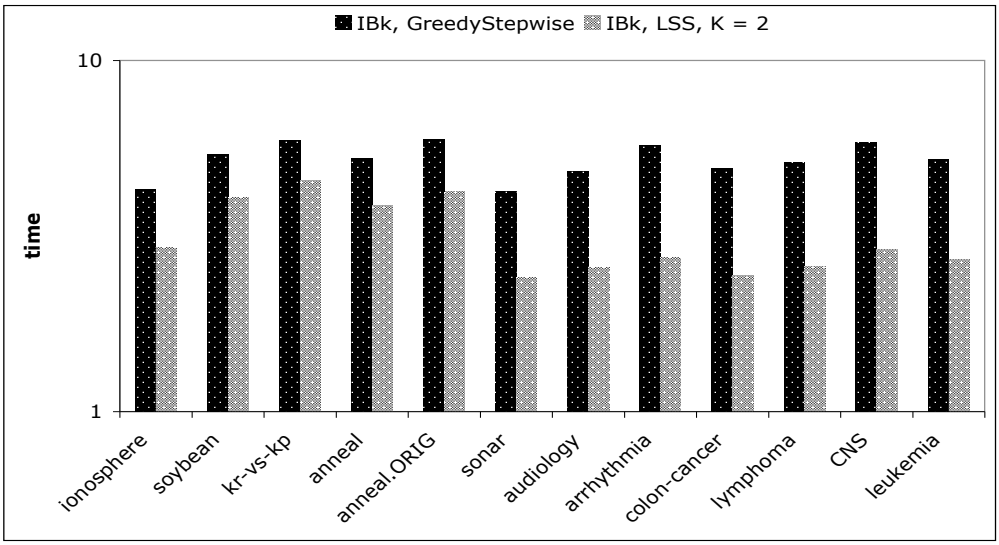


(b)

Figure 6.19: The runtime of *Linear Sequential Selection* with settings $k = 2$, 5 and 10 using J48.

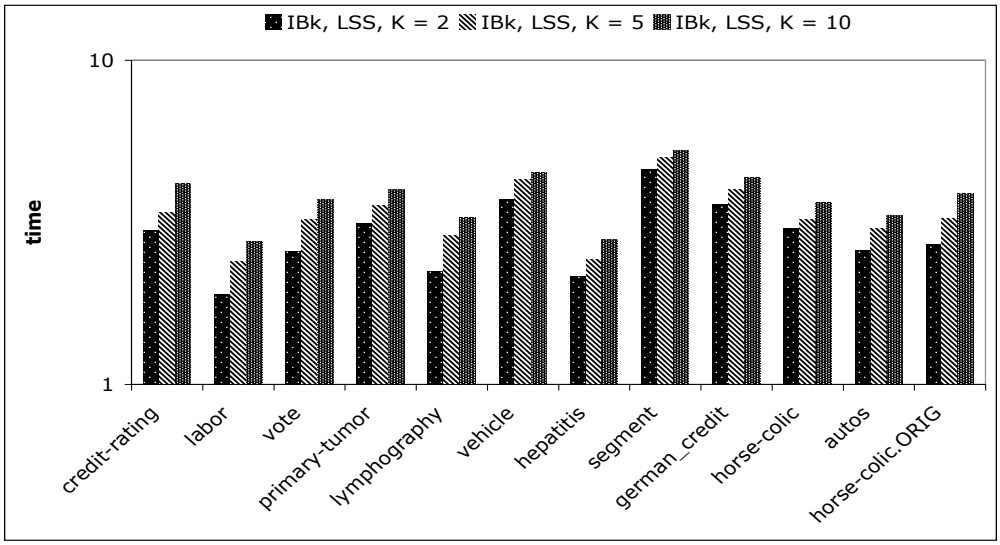


(a)

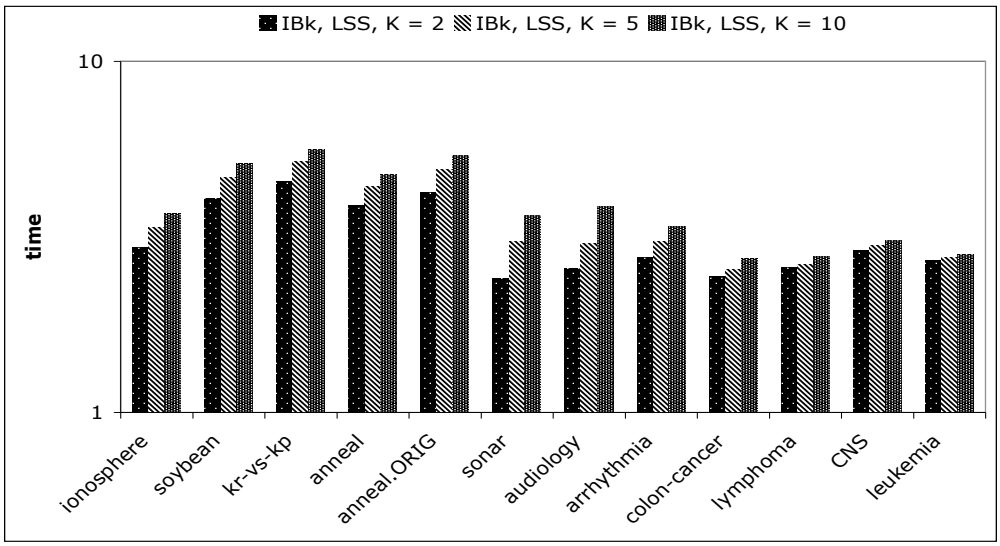


(b)

Figure 6.20: The runtime of GreedyStepwise and *Linear Sequential Selection* with default setting (i.e. $k = 2$) using IBk.



(a)



(b)

Figure 6.21: The runtime of *Linear Sequential Selection* with settings $k = 2$, 5 and 10 using IBk.

Figure 6.11 compares the runtime of naive Bayes using different settings of k in *Linear Sequential Selection*. The values of k in *Linear Sequential Selection* are set to $k = 2$, $k = 5$ and $k = 10$. The figure shows that the speed becomes slower with increase in the value of k on 15 of 24 datasets.

Time using J48

Figure 6.18 shows the runtime of J48 using GreedyStepwise and *Linear Sequential Selection* with default setting (i.e. $k = 2$). Note that the runtime is sum of feature selection time and classification time and it is expressed on a logarithmic scale. The figure shows that the runtime of J48 using *Linear Sequential Selection* is faster than j48 using GreedyStepwise on all 24 datasets.

Figure 6.13 compares the runtime of J48 using different settings of k in *Linear Sequential Selection*. The values of k in *Linear Sequential Selection* are set to $k = 2$, $k = 5$ and $k = 10$. The figure shows that the speed becomes slower with increase in the value of k on all 24 datasets. The comparison of Figure 6.18 and Figure 6.19 shows that even when k in *Linear Sequential Selection* is 10, *Linear Sequential Selection* is still faster than J48 using GreedyStepwise.

Time using IBk

Figure 6.20 shows the runtime of IBk using GreedyStepwise and *Linear Sequential Selection* with default setting (i.e. $k = 2$). Note that the runtime is sum of feature selection time and classification time and it is expressed on a logarithmic scale. The figure shows that the runtime of IBk using *Linear Sequential Selection* is faster than IBk using GreedyStepwise on all 24 datasets.

Figure 6.15 compares the runtime of IBk using different settings of k in *Linear Sequential Selection*. The values of k in *Linear Sequential Selection* are set to $k = 2$, $k = 5$ and $k = 10$. The figure shows that the speed becomes slower with increase in the value of k on all 24 datasets. The comparison of Figure 6.20 and Figure 6.21 shows that even when k in *Linear Sequential Selection* is 10, *Linear Sequential Selection* is still faster than IBk using GreedyStepwise on most of the datasets.

6.1.4 Discussion

From above results, we can see that all 3 ML algorithms are faster using *Linear Sequential Selection* than using GreedyStepwise with lower value of k. However, the speed becomes slower with the increase in the value of k. Our final experiment is done with $k = 10$ in *Linear Sequential Selection*. J48 with that setting of *Linear Sequential Selection* is still faster than GreedyStepwise on all datasets but naive Bayes and IBk are slower in some datasets, mostly low-dimension datasets. This is because *Linear Sequential Selection* searches only top k features at each iteration rather than all n features. Besides, the search process stops when addition of features do not improve accuracy. This might help the search to stop earlier rather than going into the opposite end of the search space.

The number of features selected by *Linear Sequential Selection* is less than those selected by GreedyStepwise for the majority of datasets. One of the reasons is the ranking of the features before the search progress. Information gain is used for ranking all the features and then, the features are sorted. This will stop the search procedure to interact with the irrelevant features earlier. However, the number increases with the increase in the value of k.

In terms of accuracy, all 3 ML algorithms using *Linear Sequential Selection* with $k = 2$ are less accurate than without feature selection and with GreedyStepwise for half of the datasets. However, the accuracy increases with the increase in the value of k in *Linear Sequential Selection*. The highest setting of k for our experiment is 10. On that setting in *Linear Sequential Selection*, all 3 ML algorithms maintain or show improvement in the accuracy from without feature selection and with GreedyStepwise. Though this has compromised the speed mostly in the low-dimension datasets.

6.2 Results – Randomized Linear Sequential Selection

This section shows the performance of the *Randomized Linear Sequential Selection* using 3 ML algorithms. In *Randomized Linear Sequential Selection*, k represents top k feature set and m represents number of features in the top k feature set to be replaced by randomly selected features. Three different values of k (2, 5 and 10) are used. With $k = 2$, the

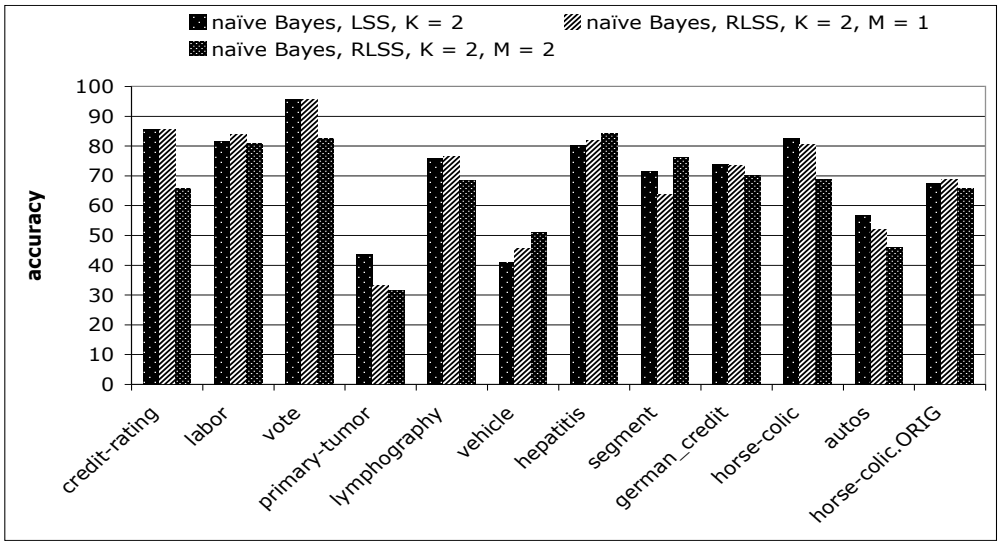
experiments are conducted by setting $m = 1$ and 2. With $k = 5$, m is 1 and 3. And with $k = 10$, m is 1, 3 and 5. The obtained results are compared with *Linear Sequential Selection* with a corresponding setting of k as in *Randomized Linear Sequential Selection.*

Classification accuracy of naive Bayes

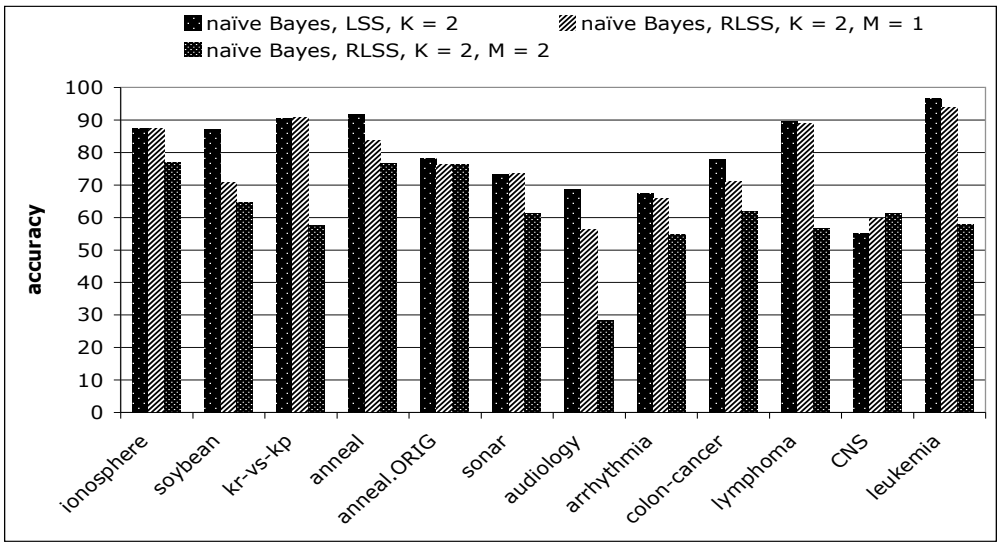
Figure 6.22 compares classification accuracy of naive Bayes using *Linear Sequential Selection* and *Randomized Linear Sequential Selection* with $k = 2$ on 24 datasets. The 2^{nd} and 3^{rd} bar of each dataset correspond to the *Randomized Linear Sequential Selection* with $m = 1$ and $m = 2$ respectively. When both k and m are 2, the classification accuracy is based on all randomly selected features. Both *Randomized Linear Selection* schemes maintain or show decrease in the accuracy on majority of the dataset. This can be seen on 15 of 24 datasets. On 3 datasets (vehicle, hepatitis and CNS), the accuracy improved. On 5 datasets (labor, lymphography, horse-colic.ORIG, kr-vs-kp and sonar), only *Randomized Linear Sequential Selection* with $m = 1$ shows a slight increase in the accuracy while on 1 dataset (segment), only *Randomized Linear Sequential Selection* with $m = 2$ shows increase in the accuracy. Between the 2 schemes of *Randomized Linear Sequential Selection*, *Randomized Linear Sequential Selection* with $m = 2$ performs quite poorly. 20 of 24 datasets show lower accuracy.

Figure 6.23 compares classification accuracy of naive Bayes using *Linear Sequential Selection* and *Randomized Linear Sequential Selection* with $k = 5$ on 24 datasets. The 2^{nd} and 3^{rd} bar of each dataset correspond to the *Randomized Linear Sequential Selection* with $m = 1$ and $m = 3$ respectively. Both *Randomized Linear Selection* schemes maintain or show decrease in the accuracy on 14 of 24 datasets. On 5 datasets (vote, lymphography, vehicle, ionosphere and sonar), the accuracy slightly improved. On 3 datasets (primary-tumor, soyabean and arrhythmia), only *Randomized Linear Sequential Selection* with $m = 1$ shows a slight increase in the accuracy while on 1 dataset (vote), only *Randomized Linear Sequential Selection* with $m = 3$ shows increase in the accuracy. Between the 2 schemes of *Randomized Linear Sequential Selection*, *Randomized Sequential Selection* with $m = 1$ performs better on 14 of 24 datasets.

Figure 6.24 compares classification accuracy of naive Bayes using *Linear Sequential Se-*

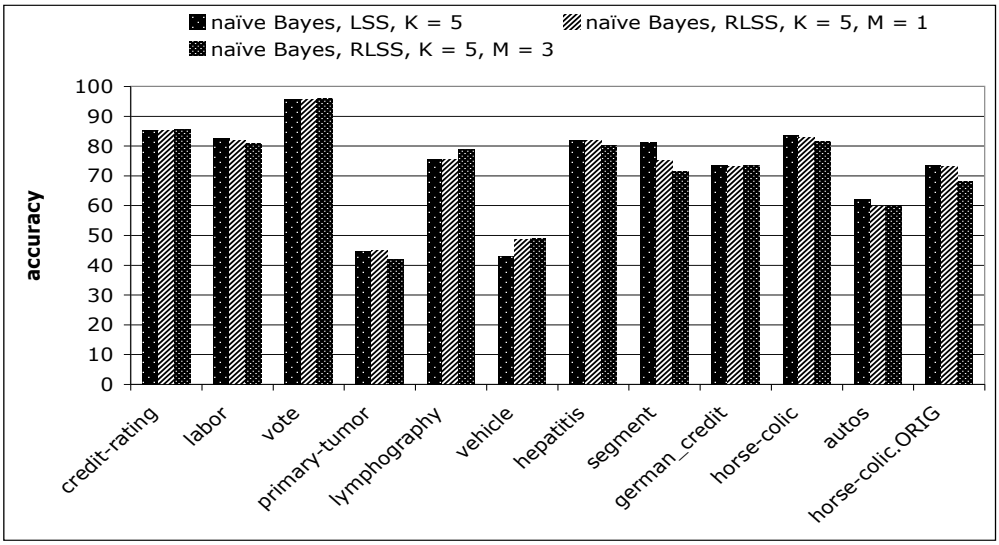


(a)

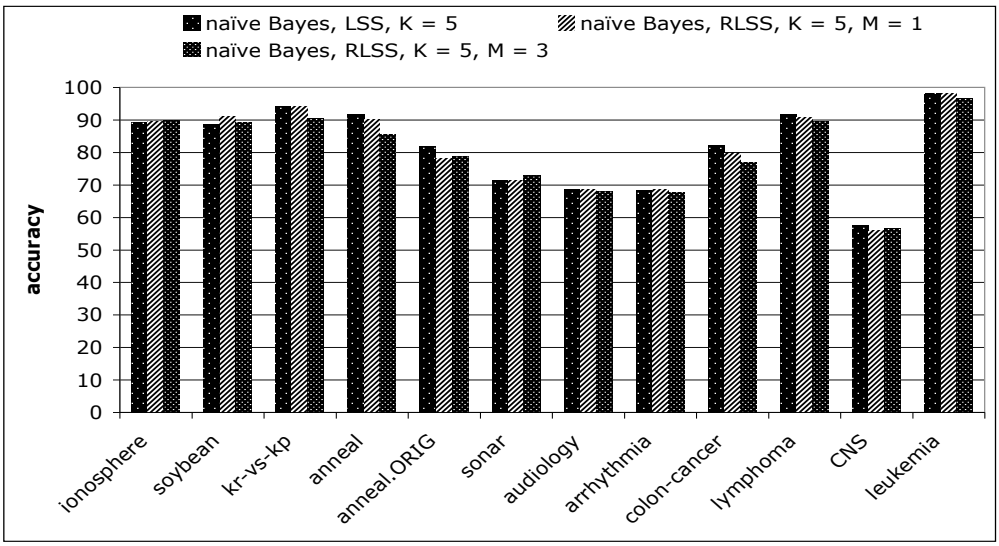


(b)

Figure 6.22: The classification accuracy of naive Bayes using *Linear Sequential Selection* and *Randomized Linear Sequential Selection*, $k = 2$, $m = 1$ and 2.

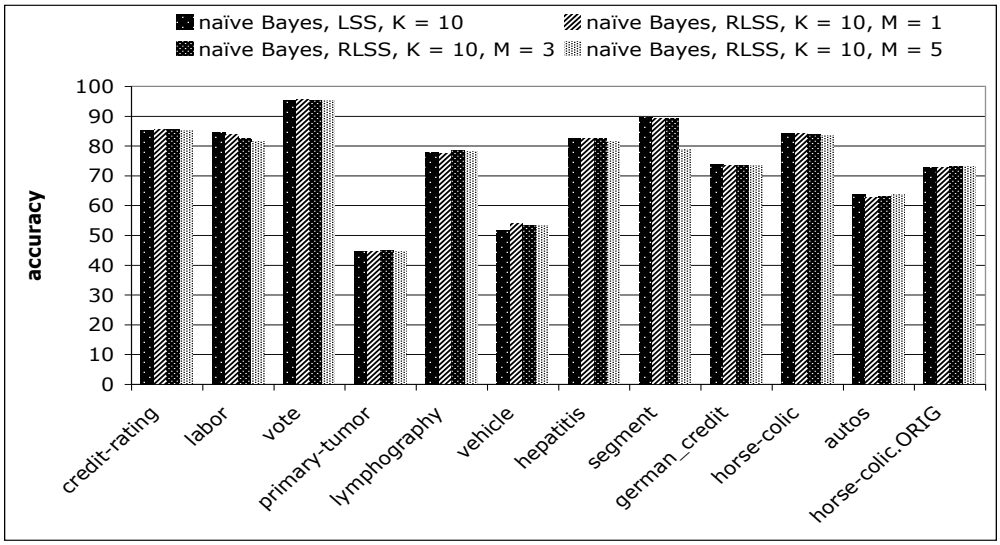


(a)

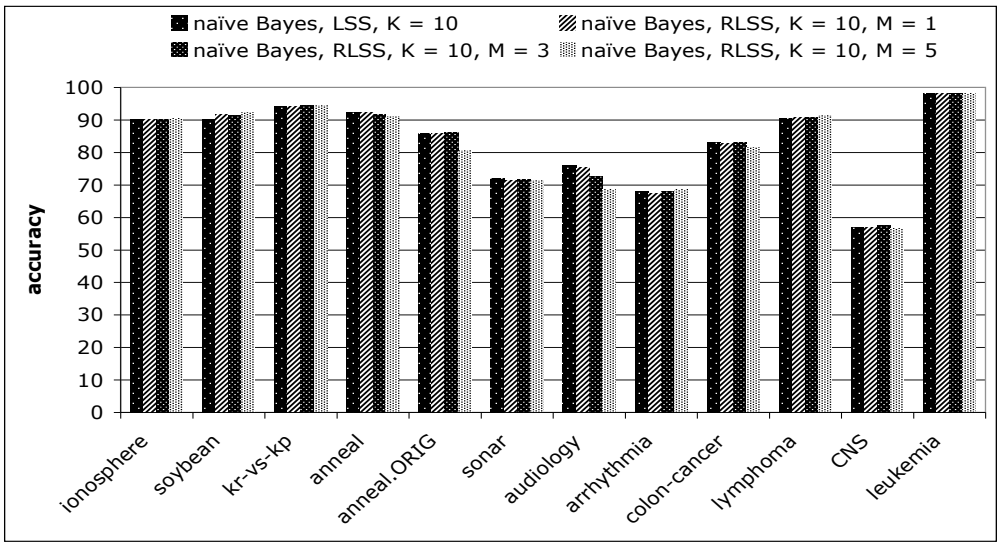


(b)

Figure 6.23: The classification accuracy of naive Bayes using *Linear Sequential Selection* and *Randomized Linear Sequential Selection*, $k = 5$, $m = 1$ and 3.



(a)



(b)

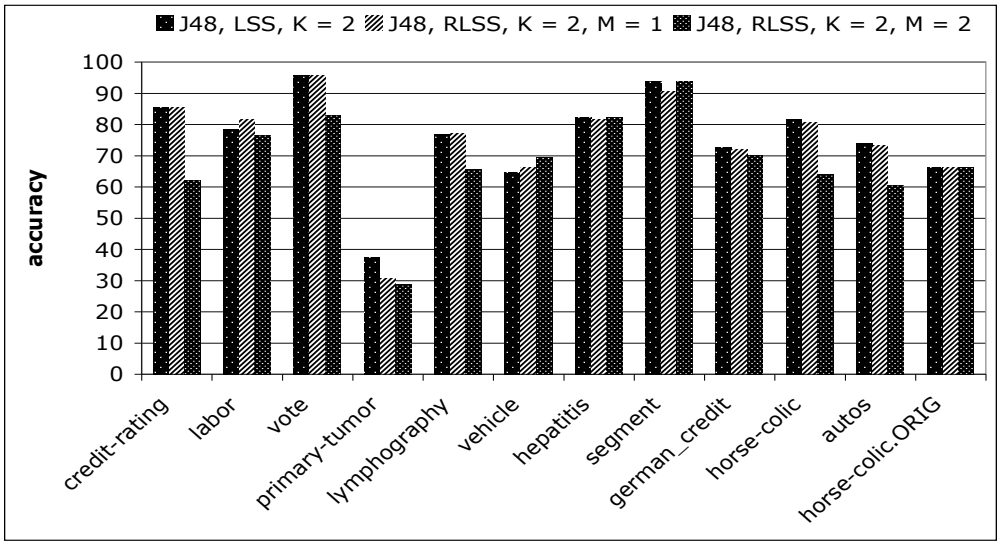
Figure 6.24: The classification accuracy of naive Bayes using *Linear Sequential Selection* and *Randomized Linear Sequential Selection*, $k = 10$, $m = 1$, 3 and 5.

lection and *Randomized Linear Sequential Selection* with $k = 10$ on 24 datasets. The 2^{nd}, 3^{rd} and 4^{th} bar of each dataset correspond to the *Randomized Linear Sequential Selection* with $m = 1$, $m = 3$ and $m = 5$ respectively. The figure shows that, except on 2 datasets (segment and anneal.ORIG), there is no difference in the classification accuracy between *Linear Sequential Selection* and *Randomized Linear Sequential Selection* and even between different schemes of *Randomized Linear Sequential Selection*, the accuracy is similar. On those datasets, *Randomized Linear Sequential Selection* with $m = 5$ shows significant decrease in the accuracy from the rest of the schemes.

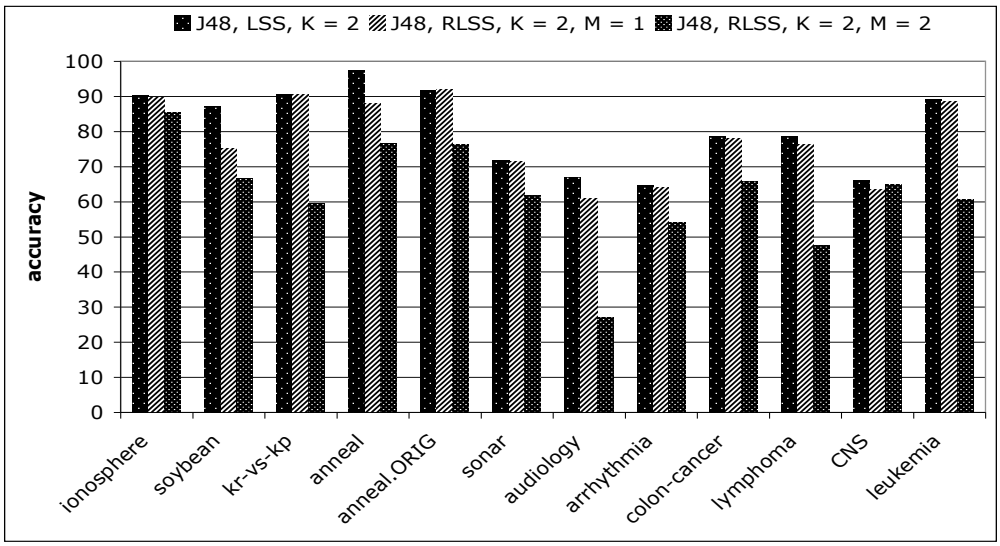
Classification accuracy of J48

Figure 6.25 compares classification accuracy of J48 using *Linear Sequential Selection* and *Randomized Linear Sequential Selection*; with $k = 2$ on 24 datasets. The 2^{nd} and 3^{rd} bar of each dataset correspond to the *Randomized Linear Sequential Selection* with $m = 1$ and $m = 2$ respectively. When both k and m are 2, the classification accuracy is based on all randomly selected features. Both *Randomized Linear Selection* schemes maintain or show decrease in the accuracy on majority of the dataset. This can be seen on 17 of 24 datasets. Only 1 dataset (vehicle) shows improved accuracy. On 4 datasets (labor, lymphography, kr-vs-kp and anneal.ORIG), only *Randomized Linear Sequential Selection* with $m = 1$ shows slight increase in the accuracy while on 2 datasets (hepatitis and segment), only *Randomized Sequential Selection* with $m = 2$ shows increase in the accuracy. Between the 2 schemes of *Randomized Linear Sequential Selection*, *Randomized Linear Sequential Selection* with $m = 2$ performs quite poorly. 20 of 24 datasets shows lower accuracy.

Figure 6.26 compares classification accuracy of J48 using *Linear Sequential Selection* and *Randomized Linear Sequential Selection* with $k = 5$ on 24 datasets. The 2^{nd} and 3^{rd} bar of each dataset correspond to the *Randomized Linear Sequential Selection* with $m = 1$ and $m = 3$ respectively. Both *Randomized Linear Selection* schemes maintain or show decrease in the accuracy on 10 of 24 datasets. On 4 datasets (vehicle, ionosphere, sonar and colon-cancer), the accuracy slightly improved. On 7 datasets (labor, primary-tumor, horse.COLIC, soyabean, kr-vs-kp, lymphoma and CNS), only *Randomized Linear Sequential Selection* with $m = 1$ shows a slight increase in the accuracy while on 2 datasets (autos

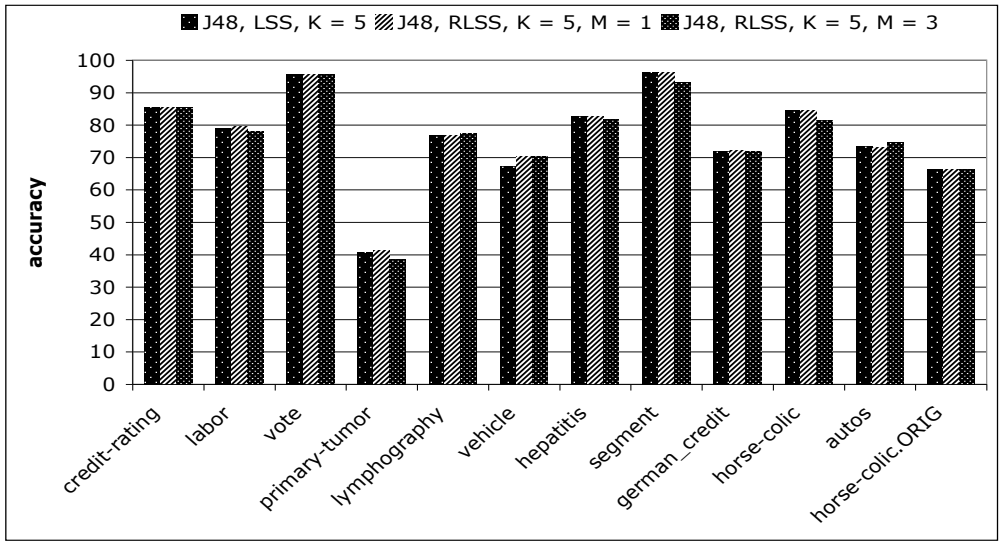


(a)

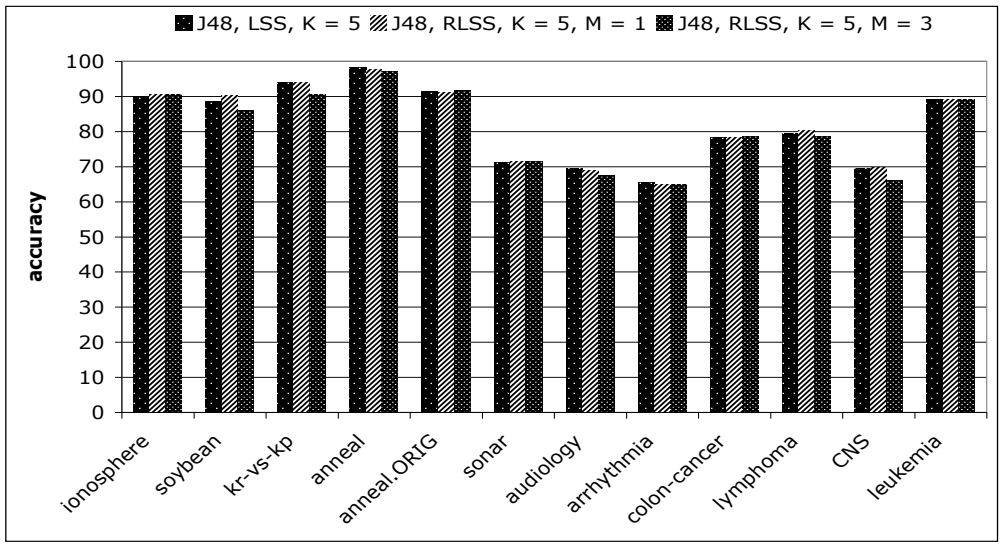


(b)

Figure 6.25: The classification accuracy of J48 using *Linear Sequential Selection* and *Randomized Linear Sequential Selection*, $k = 2$, $m = 1$ and 2.

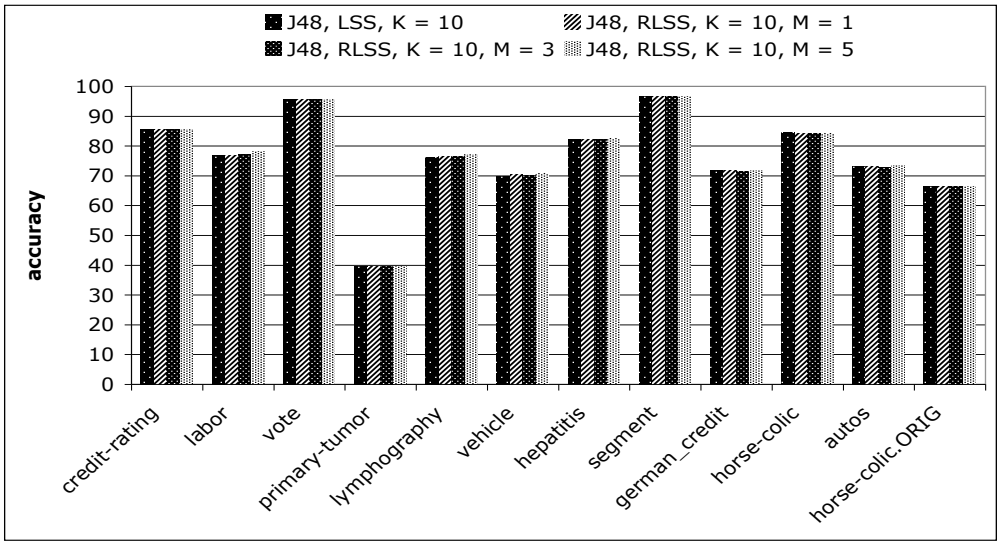


(a)

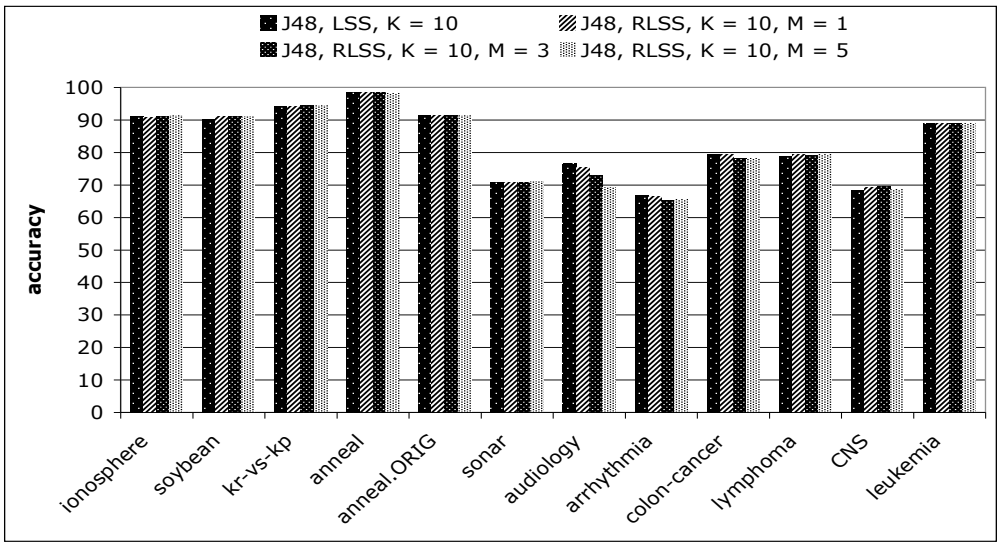


(b)

Figure 6.26: The classification accuracy of J48 using *Linear Sequential Selection* and *Randomized Linear Sequential Selection*, $k = 5$, $m = 1$ and 3.



(a)



(b)

Figure 6.27: The classification accuracy of J48 using *Linear Sequential Selection* and *Randomized Linear Sequential Selection*, $k = 10$, $m = 1$, 3 and 5.

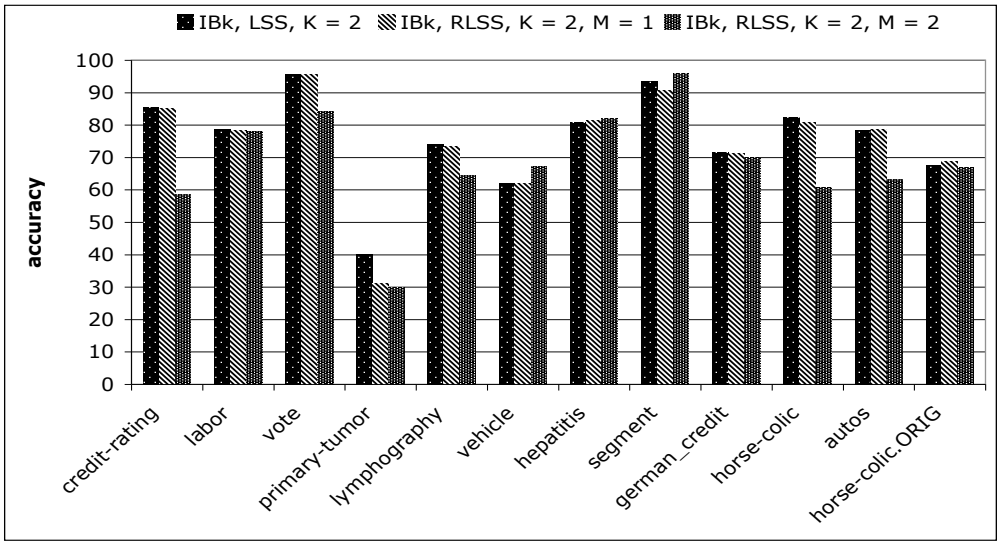
and anneal.ORIG), only *Randomized Linear Sequential Selection* with $m = 3$ shows increase in the accuracy. Between the 2 schemes of *Randomized Linear Sequential Selection*, *Randomized Sequential Selection* with $m = 1$ performs better on 18 of 24 datasets.

Figure 6.27 compares classification accuracy of J48 using *Linear Sequential Selection* and *Randomized Linear Sequential Selection* with $k = 10$ on 24 datasets. The 2^{nd}, 3^{rd} and 4^{th} bar of each dataset correspond to the *Randomized Linear Sequential Selection* with $m = 1$, $m = 3$ and $m = 5$ respectively. The figure shows that, except on 1 dataset (audiology), there is no difference in the classification accuracy between *Linear Sequential Selection* and *Randomized Linear Sequential Selection* and even between different schemes of *Randomized Linear Sequential Selection*, the accuracy is similar. On the audiology dataset, *Randomized Linear Sequential Selection* with $m = 5$ shows a decrease in accuracy compared to the other schemes.

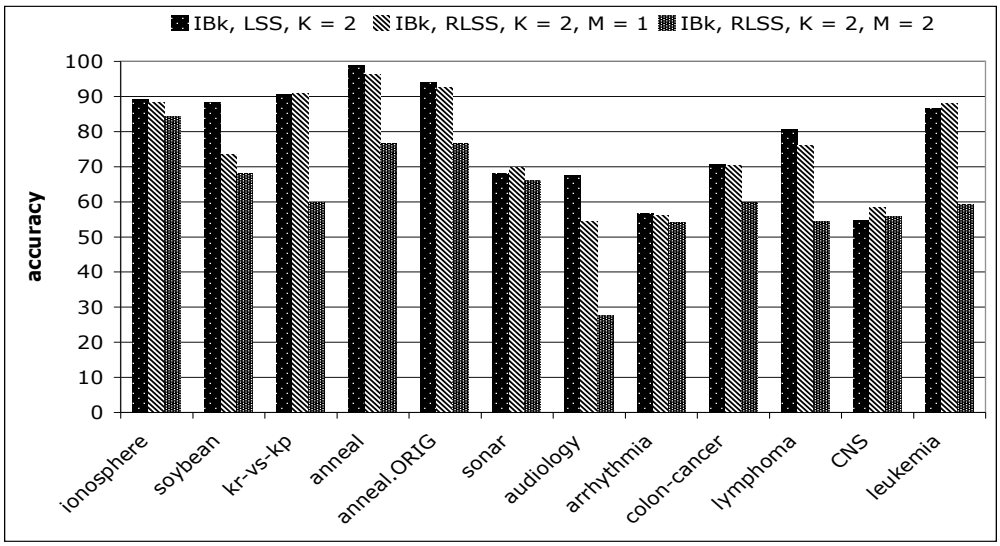
Classification accuracy of IBk

Figure 6.28 compares classification accuracy of IBk using *Linear Sequential Selection* and *Randomized Linear Sequential Selection* with $k = 2$ on 24 datasets. The 2^{nd} and 3^{rd} bar of each dataset correspond to the *Randomized Linear Sequential Selection* with $m = 1$ and $m = 2$ respectively. When both k and m are 2, the classification accuracy is based on all randomly selected features. Both *Randomized Linear Selection* schemes maintain or show decrease in the accuracy on a majority of the dataset. This can be seen on 15 of 24 datasets. Two datasets (hepatitis and CNS) show improved accuracy. On 5 datasets (autos, horse-colic.ORIG, kr-vs-kp, sonar and leukemia), only *Randomized Linear Sequential Selection* with $m = 1$ shows a slight increase in the accuracy while on 2 datasets (vehicle and segment), only *Randomized Linear Sequential Selection* with $m = 2$ shows increase in the accuracy. Between the 2 schemes of *Randomized Linear Sequential Selection*, *Randomized Linear Sequential Selection* with $m = 2$ performs quite poorly. 21 of 24 datasets show low accuracy.

Figure 6.29 compares classification accuracy of IBk using *Linear Sequential Selection* and *Randomized Linear Sequential Selection* with $k = 5$ on 24 datasets. The 2^{nd} and 3^{rd} bar of each dataset correspond to the *Randomized Linear Sequential Selection* with m

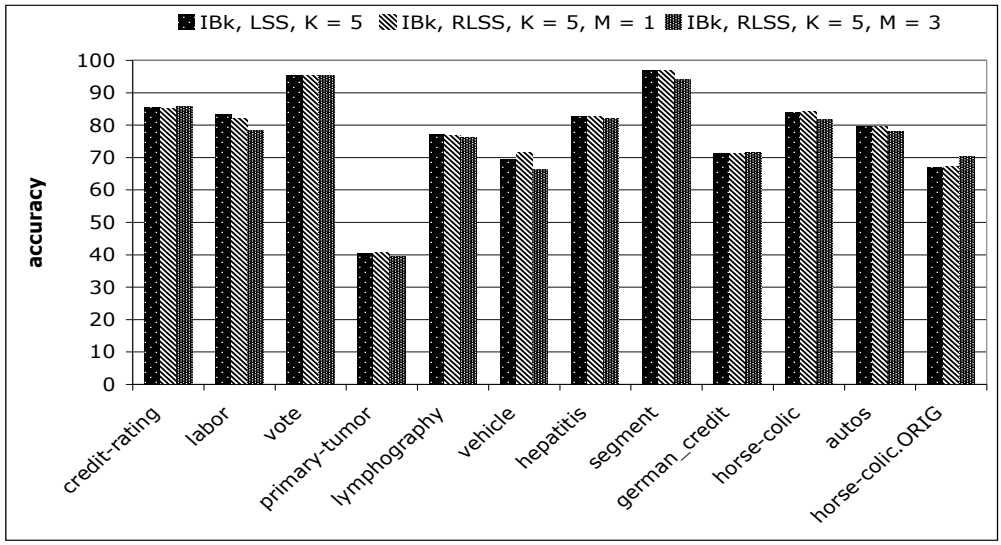


(a)

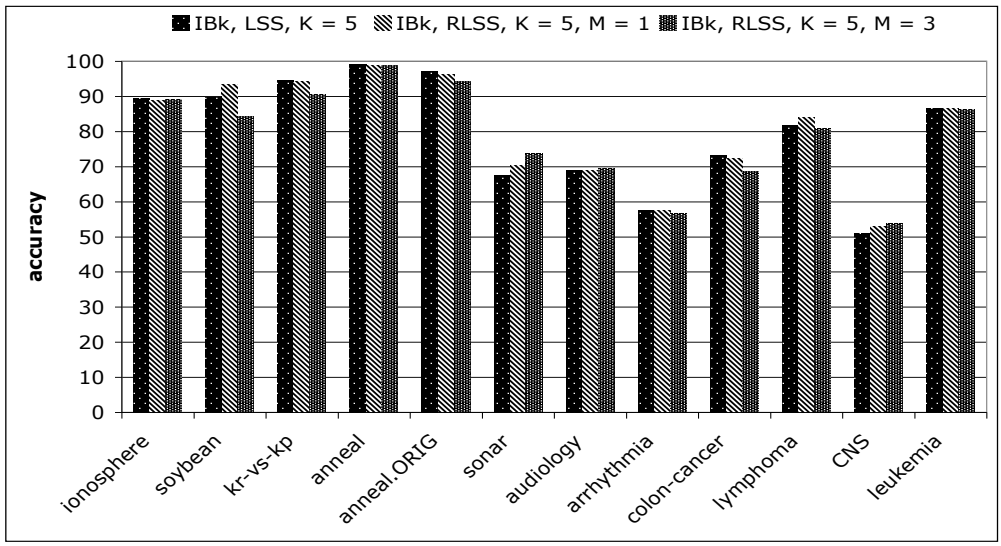


(b)

Figure 6.28: The classification accuracy of IBk using *Linear Sequential Selection* and *Randomized Linear Sequential Selection*, $k = 2$, $m = 1$ and 2.

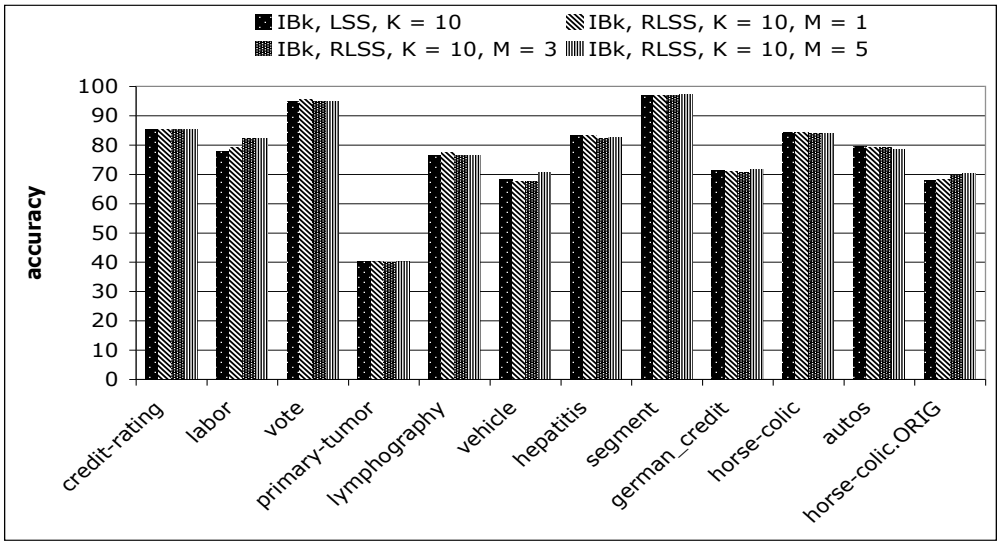


(a)

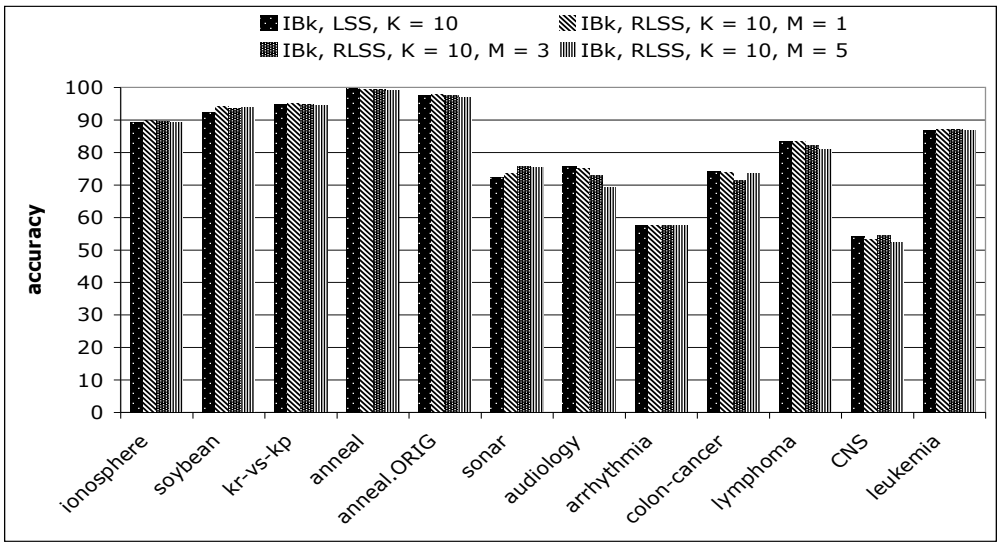


(b)

Figure 6.29: The classification accuracy of IBk using *Linear Sequential Selection* and *Randomized Linear Sequential Selection*, $k = 5$, m =1 and 3.



(a)



(b)

Figure 6.30: The classification accuracy of IBk using *Linear Sequential Selection* and *Randomized Linear Sequential Selection*, $k = 10$, $m = 1$, 3 and 5.

= 1 and $m = 3$ respectively. Both *Randomized Linear Selection* schemes maintain or show decrease in the accuracy on 11 of 24 datasets. On 5 datasets (german_credit, horse-colic.ORIG, sonar, audiology and CNS), the accuracy slightly improved. On 6 datasets (primary-tumor, vehicle, hepatitis, horse.COLIC, and lymphoma), only *Randomized Linear Sequential Selection* with $m = 1$ shows a slight increase in the accuracy while on 2 datasets (credit-rating and ionosphere), only *Randomized Linear Sequential Selection* with $m = 3$ shows increase in the accuracy. Between the 2 schemes of *Randomized Linear Sequential Selection*, *Randomized Sequential Selection* with $m = 1$ performs better on 16 of 24 datasets.

Figure 6.30 compares classification accuracy of IBk using *Linear Sequential Selection* and *Randomized Linear Sequential Selection* with $k = 10$ on 24 datasets. The 2^{nd}, 3^{rd} and 4^{th} bar of each dataset correspond to the *Randomized Linear Sequential Selection* with $m = 1$, $m = 3$ and $m = 5$ respectively. The figure shows that, except on 1 dataset (audiology), there is no difference in the classification accuracy between *Linear Sequential Selection* and *Randomized Linear Sequential Selection*, and even between different schemes of *Randomized Linear Sequential Selection*. On the audiology dataset, *Randomized Linear Sequential Selection* with $m = 5$ shows decrease in the accuracy compared to the other schemes.

6.2.1 Discussion

From above results, we can see that all 3 ML algorithms using *Randomized Linear Sequential Selection* have similar classification accuracy to using *Linear Sequential Selection* with the same value of k on most of the datasets. In *Randomized Linear Sequential Selection*, m least favorable features in top k features are replaced by randomly selected features. During the selection process, due to ranking of the features, *Randomized Linear Sequential Selection* might selects only top 1^{st} or 2^{nd} feature from the top k feature set. This causes the same classification accuracy as the *Linear Sequential Selection*. The complete randomization (i.e. $k = 2$ and $m = 2$ in *Randomized Linear Sequential Selection*) scheme has shown significant decrease in the classification accuracy.

6.3 Results – Feature Selection Ensemble

This section analyzes the results obtained from the experiments with *Feature Selection Ensemble*. The experiments are conducted by applying *Linear Sequential Selection* as the feature selection algorithm in *Feature Selection Ensemble*. Ten classifiers of the base classifier are generated for the ensemble. The prediction of the each of 10 classifiers are combined by voting as described in Chapter 4. The results are shown for only *Linear Sequential Selection* with $k = 10$ where k is top k features. Other settings of *Linear Sequential Selection* are omitted as they have shown the similar results. The omitted results are presented in Appendix C.

Classification accuracy of naive Bayes

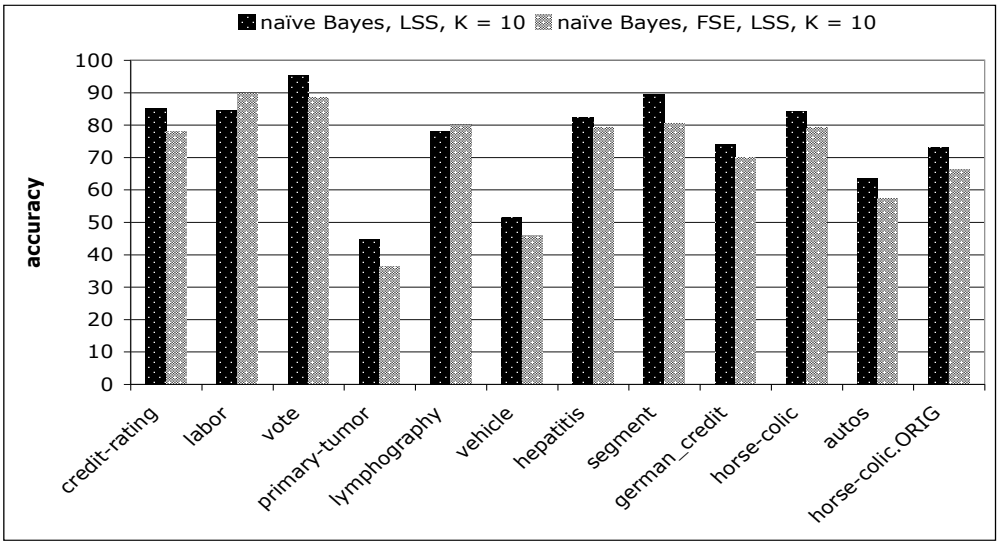
Figure 6.31 shows the classification accuracy of *Feature Selection Ensemble* using naive Bayes as the base classifier. The obtained results are compared with the accuracy of single naive Bayes with *Feature Sequential Selection*. *Feature Selection Ensemble* is able to improve accuracy on 3 of 4 microarray datasets. Three microarray datasets are colon-cancer, lymphoma and CNS. Only leukemia shows decrease in the accuracy. On 20 UCI datasets, *Feature Selection Ensemble* improves accuracy on just 2 datasets (labor and lymphography).

Classification accuracy of J48

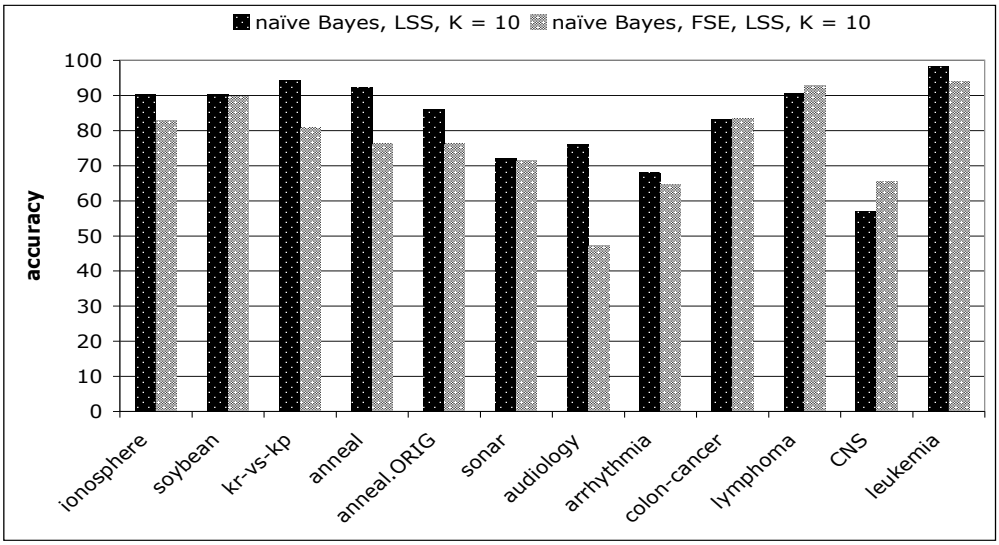
Figure 6.32 shows the classification accuracy of *Feature Selection Ensemble* using J48 as the base classifier. The obtained results are compared with the accuracy of single J48 using *Feature Sequential Selection*. *Feature Selection Ensemble* is able to improve accuracy on 3 of 4 microarray datasets. These microarray datasets are colon-cancer, lymphoma and leukemia. Only CNS a shows decrease in the accuracy. On 20 UCI datasets, *Feature Selection Ensemble* improves accuracy on just 4 datasets (lymphography, vehicle, autos, soybean and sonar).

Classification accuracy of IBk

Figure 6.33 shows the classification accuracy of *Feature Selection Ensemble* using IBk as the base classifier. The obtained results are compared with the accuracy of single IBk using

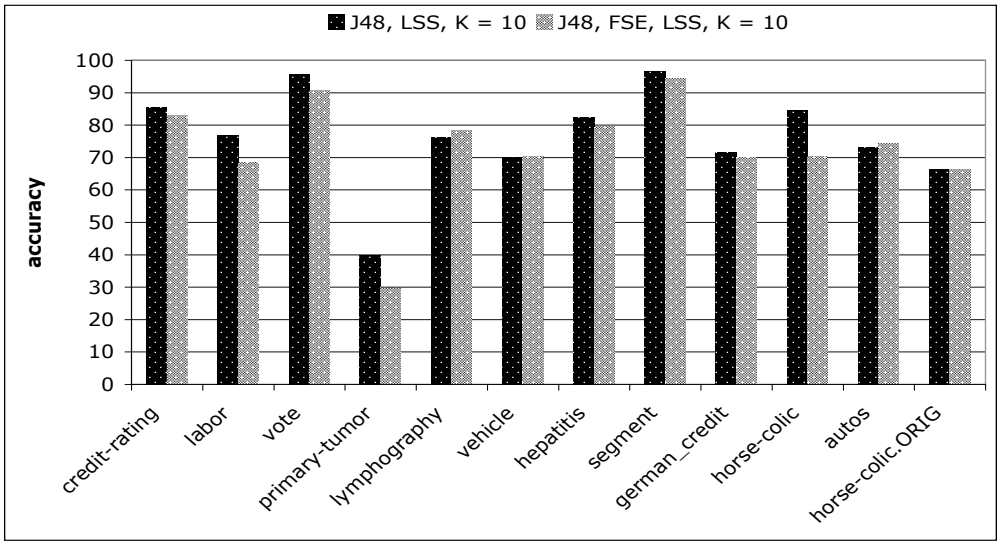


(a)

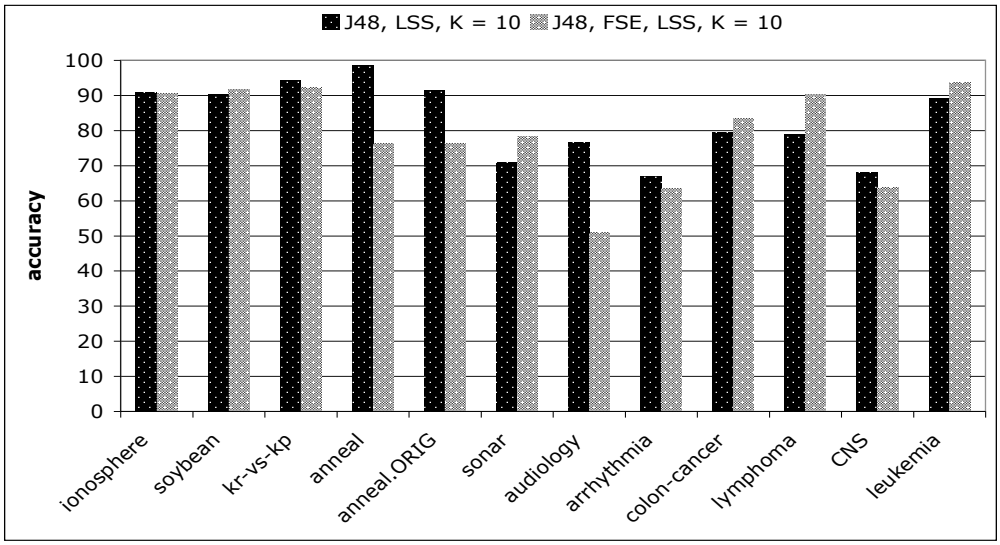


(b)

Figure 6.31: The accuracy of naive Bayes using *Linear Sequential Selection* and *Feature Selection Ensemble* using naive Bayes as the base classifier.

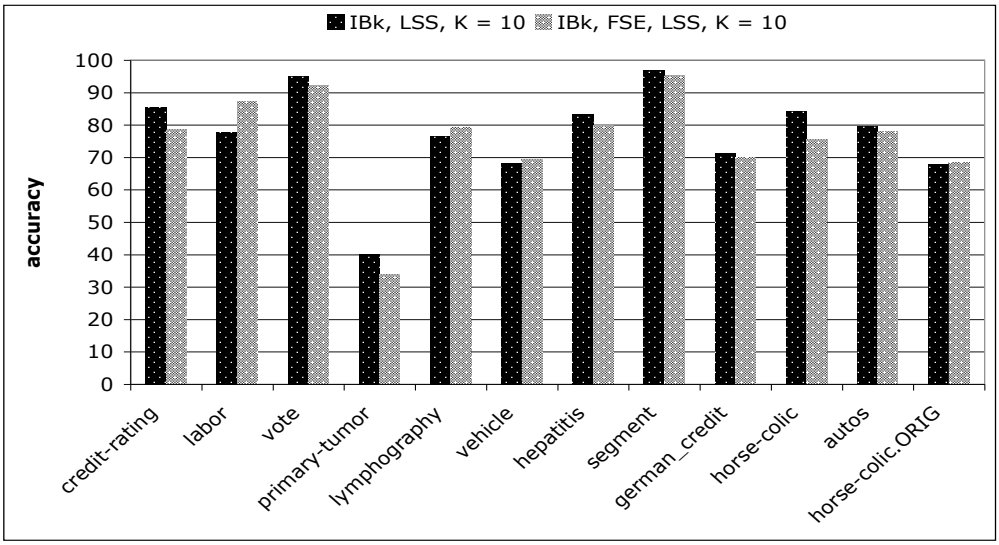


(a)

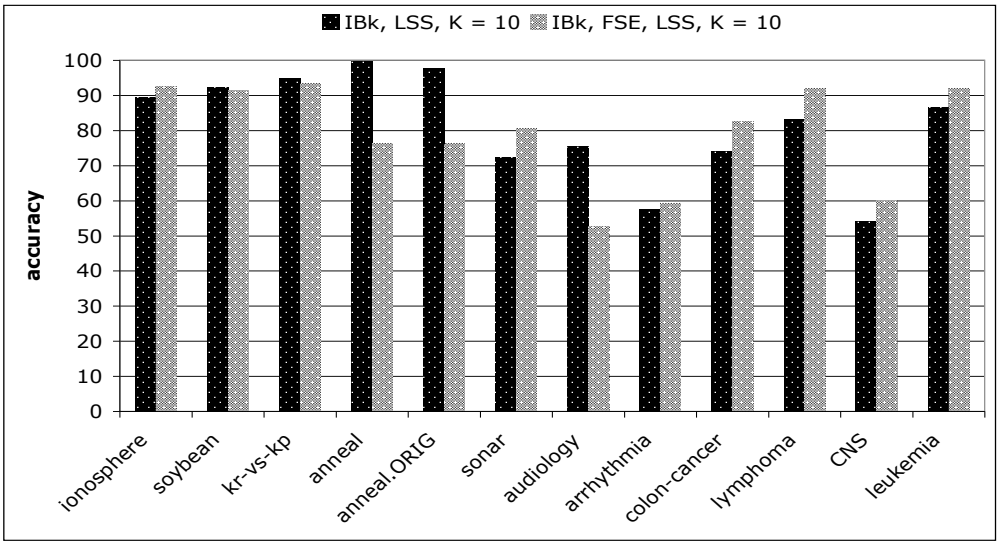


(b)

Figure 6.32: The accuracy of J48 using *Linear Sequential Selection* and *Feature Selection Ensemble* using J48 as the base classifier.



(a)



(b)

Figure 6.33: The accuracy of IBk using *Linear Sequential Selection* and *Feature Selection Ensemble* using IBk as the base classifier.

Feature Sequential Selection. *Feature Selection Ensemble* is able to improve accuracy on all 4 datasets (colon-cancer, lymphoma, CNS and leukemia) where as on 20 UCI datasets, *Feature Selection Ensemble* improves accuracy on just 6 datasets (labor, lymphography, vehicle, horse-colic.ORIG, ionosphere and sonar).

6.3.1 Discussion

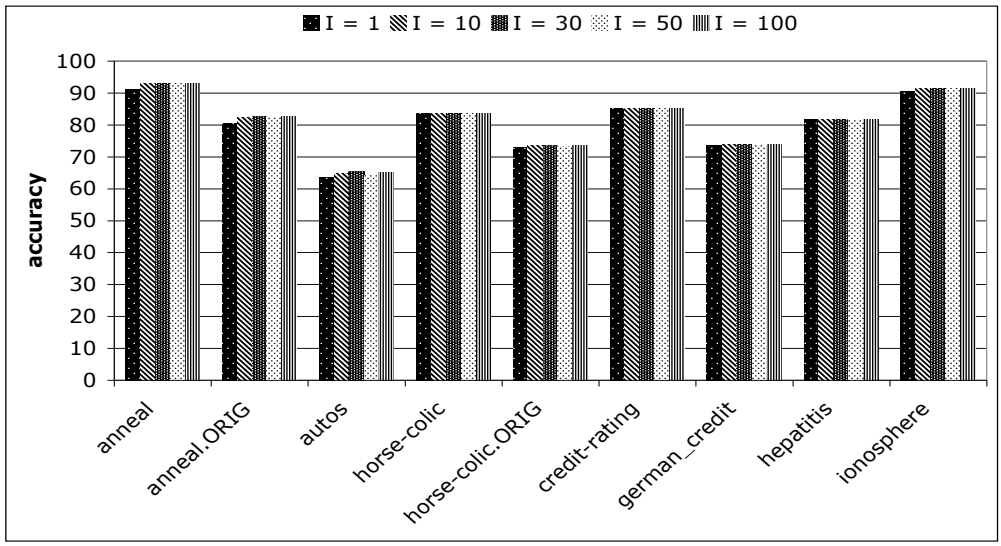
From the results above, we can see that *Feature Selection Ensemble* does not perform well with the dataset with small number of features. On those datasets, *Feature Selection Ensemble* could not able to generate 10 classifiers for predictions due to lack of enough features. This can be the reason for the low classification accuracy. However, *Feature Selection Ensemble* performs better with the dataset with large number of features mostly microarray datasets which is encouraging. On microarray datasets, *Feature Selection Ensemble* is able to generate minimum number of features i.e. 10. Since each classifier is built from the different subset of features, diversity is accomplished. This leads to the improved accuracy.

6.4 Results – Random Feature Ensemble

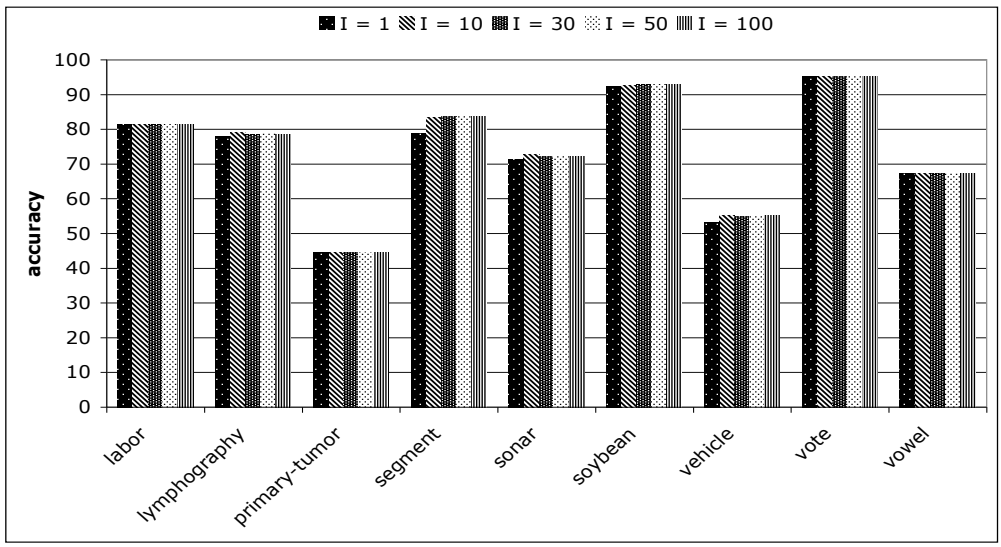
As mentioned earlier in Chapter 4, *Random Feature Ensemble* applies *Randomized Feature Sequential Selection* for feature selection. The experiments are conducted by setting $k = 10$ and $m = 5$ in the *Randomized Feature Sequential Selection*, where k is top k features and m is the number of least favorable feature in the top k feature to be replaced by randomly selected features. Four *Random Feature Ensembles* of each of 3 ML algorithms are created by varying the number of classifiers for comparison. Each of them contains 10, 30, 50 and 100 classifiers. The results obtained are compared to the respective single classifier.

Classification accuracy of naive Bayes

Figure 6.34 compares the classification accuracy of single naive Bayes using *Randomized Linear Sequential Selection* and 4 *Random Feature Ensembles* created as described above using naive Bayes as the base classifier on 18 datasets. The figure shows that all 4 *Random*

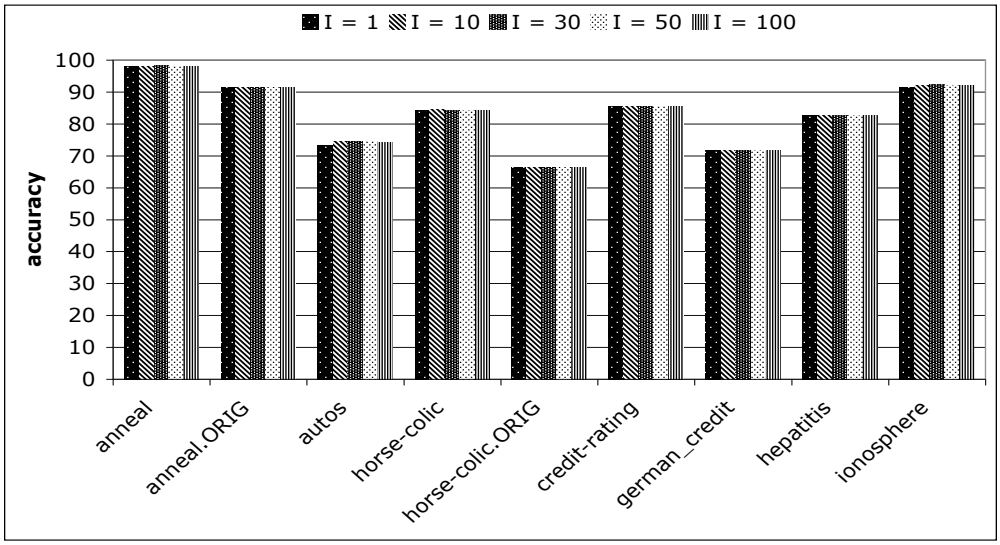


(a)

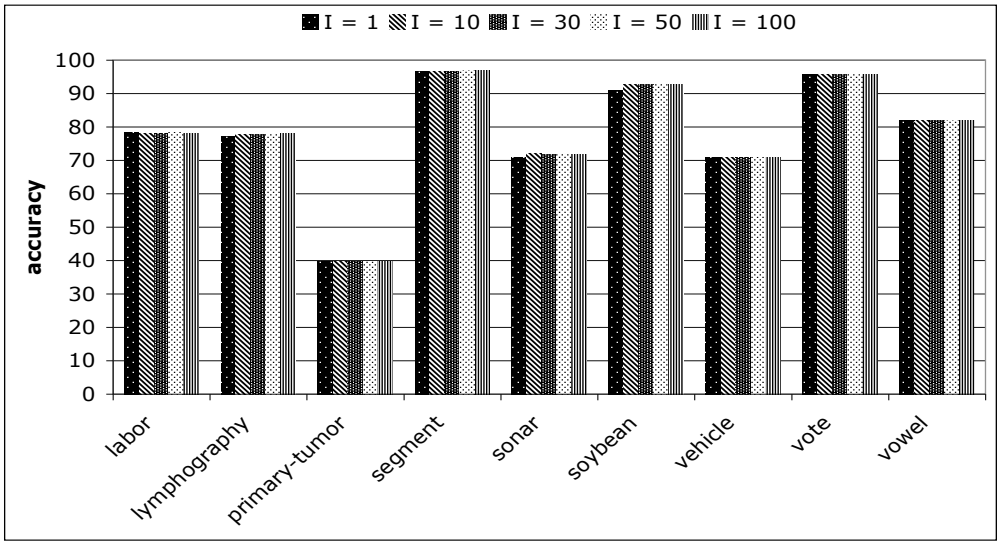


(b)

Figure 6.34: The classification accuracy of *Random Feature Ensemble* using naive Bayes as the base classifier.

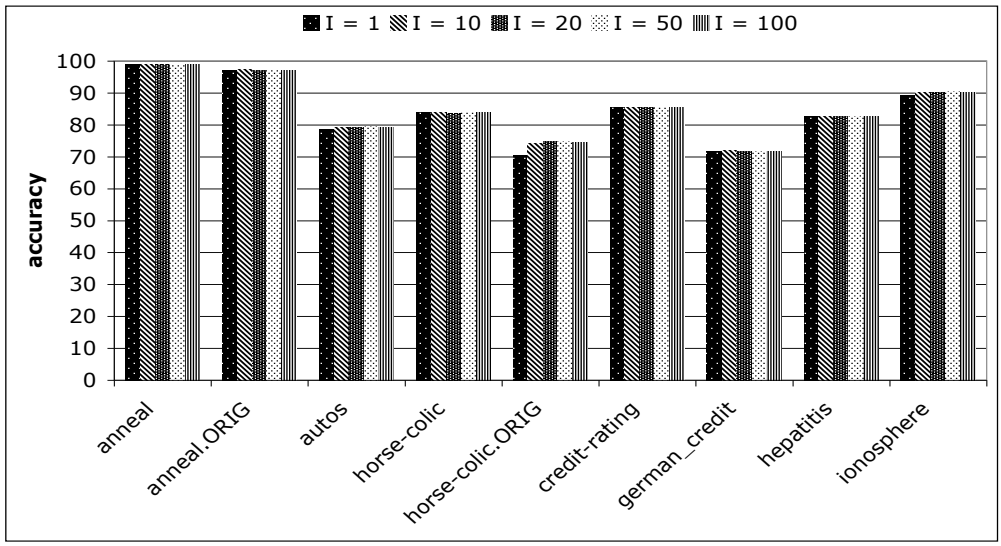


(a)

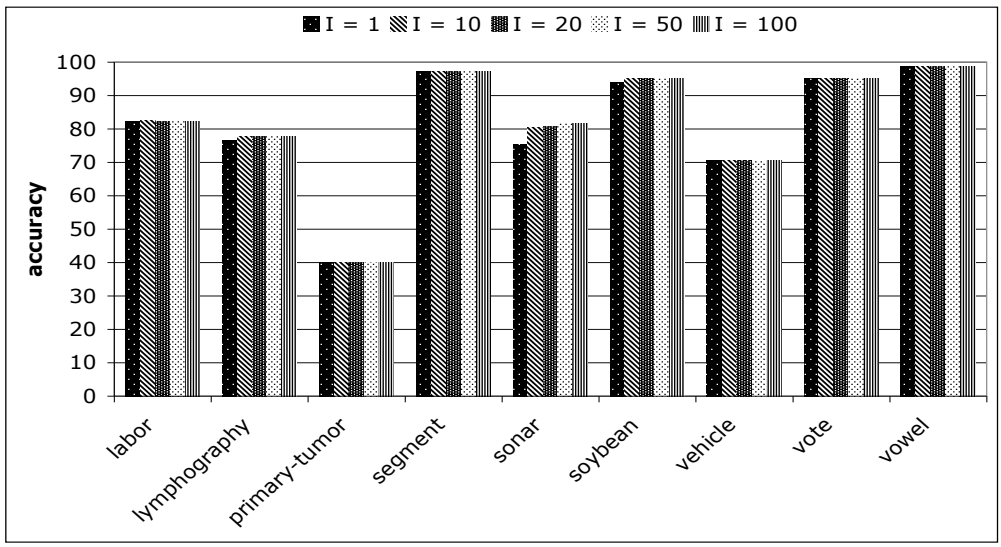


(b)

Figure 6.35: The classification accuracy of *Random Feature Ensemble* using J48 as the base classifier.



(a)



(b)

Figure 6.36: The classification accuracy of *Random Feature Ensemble* using IBk as the base classifier.

Feature Ensembles maintain or show slight improvement in the accuracy from the single naive Bayes. Between the 4 *Random Feature Ensembles*, the classification accuracy is similar to each other.

Classification accuracy of J48

Figure 6.35 compares the classification accuracy of single J48 using *Randomized Linear Sequential Selection* and 4 *Random Feature Ensembles* created as described above using J48 as the base classifier on 18 datasets. The figure shows that all 4 *Random Feature Ensembles* maintain or show slight improvement in the accuracy from the single J48. Between the 4 *Random Feature Ensembles*, the classification accuracy is similar to each other.

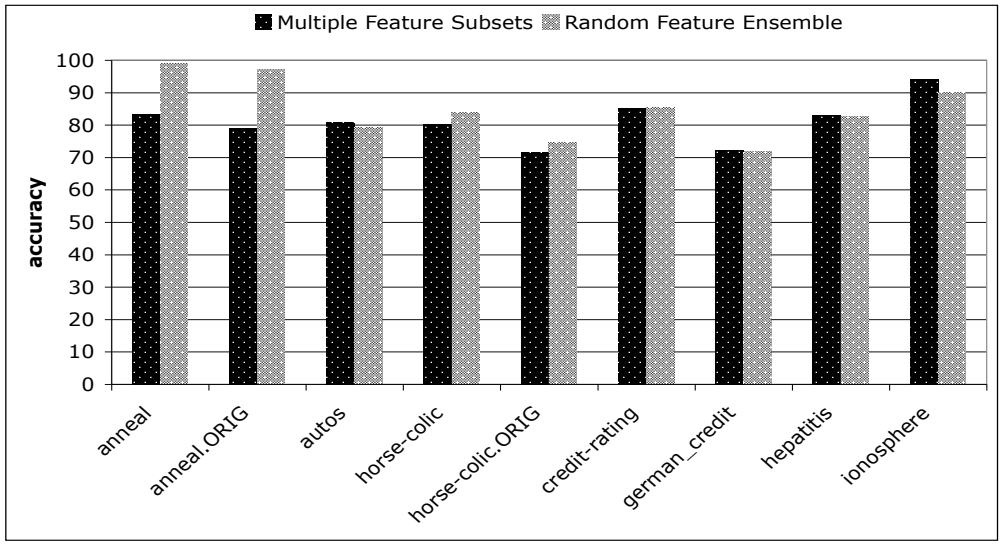
Classification accuracy of IBk

Figure 6.36 compares the classification accuracy of single IBk using *Randomized Linear Sequential Selection* and 4 *Random Feature Ensembles* created as described above using IBk as the base classifier on 18 datasets. The figure shows that all 4 *Random Feature Ensembles* maintain or show slight improvement in the accuracy from the single IBk. Between the 4 *Random Feature Ensembles*, the classification accuracy is similar to each other.

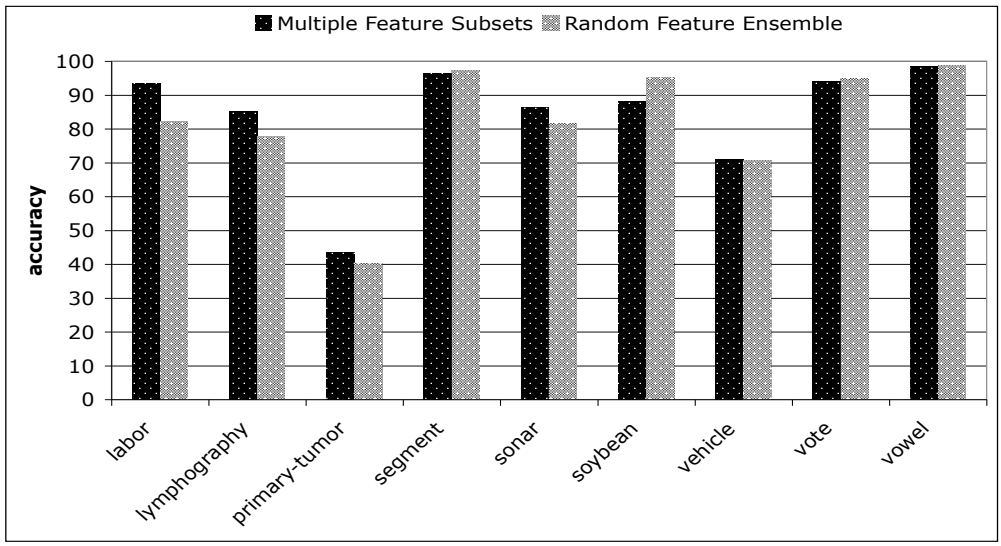
Comparison of *Random Feature Ensemble* with *Multiple Feature Subsets*

The performance of *Random Feature Ensemble* is also compared with *Multiple Feature Subsets*. *Multiple Feature Subsets* is discussed in detail in Chapter 3. *Multiple Feature Subsets* is implemented in WEKA [13] framework. Both ensembles contain 100 classifiers. IBk is used as the base classifier. The small modification has been made in *Multiple Feature Subsets*. Instead of using cross-validation to find number of features in each individual classifier in the ensemble, it is set as an user defined option. For this experiment, we set the number of features in each individual classifier to 5.

Figure 6.37 compares the classification accuracy of *Multiple Feature Subsets* and *Random Feature Ensemble* using IBk as the base classifier on 18 datasets. The figure shows that *Random Feature Ensemble* maintains or shows improvement in the accuracy on 9 datasets.



(a)



(b)

Figure 6.37: The classification accuracy of *Multiple Feature Subsets* and *Random Feature Ensemble* using IBk as the base classifier.

The remaining 9 datasets show degradation in the accuracy.

6.4.1 Discussion

From the above results, we can see that although all 3 ML algorithms using *Random Feature Ensemble* are able to improve classification accuracy slightly from the single classifier, but not significantly. Between the 4 *Random Feature Ensembles*, there is no difference between them in terms of their prediction ability. One of the reason can be the use of *Randomized Linear Sequential Selection* for feature selection. The feature selected by *Randomized Linear Sequential Selection* for each classifier is more or less similar. This may be due to the fact that *Randomized Linear Sequential Selection* replaces m bottom least favorable features from the k feature set. During the selection process, *Randomized Linear Sequential Selection* might only selects features from the top k feature set rather than randomly replaced features and the diversity between the classifiers is limited.

The comparison with *Multiple Feature Subset* has shown that *Random Feature Ensemble* is better on half of the datasets.

Chapter 7
Conclusion

We have described four algorithms for this thesis. Two, *Linear Sequential Selection* and *Randomized Linear Sequential Selection*, are feature selection methods and two, *Feature Selection Ensemble* and *Random Feature Ensemble*, are ensemble creation methods. We have implemented all four algorithms to this end and experimented with several real life datasets using three ML algorithms. The results gathered so far are satisfactory.

Linear Sequential Selection starts with ranking all the features and searches only the best k features at each step of the selection process, rather than all remaining n features. The results showed that *Linear Sequential Selection* is faster than GreedyStepwise. Even the number of features selected by *Linear Sequential Selection* is less compared to GreedyStepwise. In terms of accuracy, performance was significantly worse in only a few datasets compared to no feature selection or using GreedyStepwise. Furthermore, accuracy generally improves with higher values of k.

Randomized Linear Sequential Selection is designed to overcome the local minima problem faced by *Linear Sequential Selection*. The results showed that there is not much difference between *Linear Sequential Selection* and *Randomized Linear Sequential Selection*. *Randomized Linear Sequential Selection* usually performed as good as the *Linear Sequential Selection*. However, the result for a completely randomized selection process is far from satisfactory.

Feature Selection Ensemble is a new approach in ensemble creation methods that uses

the feature selection algorithm to create individual classifiers for the ensemble. The results showed that *Feature Selection Ensemble* performs better with the high-dimensional datasets. However, the results for datasets having fewer features are not satisfactory. The reason might be not having enough features to generate a reasonable number of classifiers for the ensemble.

Random Feature Ensemble is designed to take advantage from the feature generation scheme used by *Randomized Linear Sequential Selection*. Due to the introduction of randomization in the *Randomize Linear Sequential Selection*, it will be able to generate different feature subsets for each individual classifier for the ensemble. *Random Feature Ensemble* has shown slight improvement from the single classifier in terms of accuracy. Between 4 different *Random Feature Ensembles* (i.e. with 10, 30, 50 and 100 classifiers respectively), the accuracies are similar. This shows that these methods do not have any effect in the accuracy.

Directions for future work are as follows: one of the work involves investigating best value of k in *Linear Sequential Selection* and *Randomized Linear Sequential Selection*. The results have shown that the value of k plays a significant role in both algorithms. Other work can be to use different ranking methods. It will be interesting to see how both algorithms perform with different ranking methods. Furthermore, more future work is required on evaluating both algorithms on larger datasets.

In *Feature Selection Ensemble*, *Linear Sequential Selection* is used as the feature selection algorithm. It has shown improvement in the accuracy in the high-dimensional datasets. Besides *Linear Sequential Selection*, any other feature selection algorithm can be used for feature selection. Investigation with other methods are left for future.

Another idea for investigation is to use several feature selection methods to generate separate classifier for the ensemble. Since different feature selection methods use different biases to generate features, the feature generated by each methods can be different. This will lead to the generation of diverse classifiers for the ensemble.

Bibliography

[1] D. Aha and D. Kibler. Instance-based learning algorithms. *Machine Learning*, vol. 6, pp. 37-66, 1991.

[2] Ash A. Alizadeh, Michael B. Eisen, R. Eric Davis, Chi Ma, Izidore S. Lossos, Andreas Rosenwald, Jennifer C. Boldrick, Hajeer Sabet, Truc Tran, Xin Yu, et al. Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, vol. 403, no. 6769, pp. 503-511, 2000.

[3] H. Almuallim and T. G. Dietterich. Learning with many irrelevant features. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, pp. 547-542, MIT Press, 1991.

[4] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering of tumor and normal colon tissues probed by oligonucleotide arrays. In *Proceedings of the National Academy of Sciences*, vol. 96, pp. 6745-6750, USA, 1999

[5] D. W. Aha and R. L. Bankert. A comparative evaluation of sequential feature selection algorithms. In Proceeding of the Fifth International Workshop on Artificial Intelligence and Statistics, pp.1-7, 1995.

[6] Stephen D. Bay. Combining nearest neighbor classifiers through multiple feature subsets. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pp. 37-45, Morgan Kaufmann, 1998.

[7] Michael Bensch, Michael Schroder, Martin Bogdan and Wolfgang Rosenstiel. Feature selection for high-dimensional industrial data. In *ESANN*, pages 375-380, 2005.

[8] C. Blake, E. Keogh and C. Merz. UCI repository of machine learning databases [http://www.ics.uci.edu/mlearn/mlrepository.html]. Technical report, Department of Information and Computer Science, University of California, Irvine, CA.

[9] Avrim Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 1997.

[10] L. Breiman. Bagging Predictors. *Machine Learning*, vol. 24, pp. 123-140, 1996.

[11] R. Caruana and D. Freitag. Greedy attribute selection. In *Proceedings of the Eleventh Internation Conference on Machine Learning*, Morgan Kaufmann, 1994.

[12] K. Clarkson. An algorithm for approximate closest point queries. In *Proceedings of the Tenth Annual ACM Symposium on Computational Geometry*, pp. 160-164, 1994.

[13] Eibe Frank, Mark A. Hall, Geoffrey Holmes, Richard Kirkby and Bernhard Pfahringer. Weka - a machine learning workbench for data mining. In *The Data Mining and Knowledge Discovery Handbook*, pp. 1305-1314, Springer, 2005.

[14] D. Gamberger and N. Lavrac. Conditions for Occams Razor applicability and noise elimination. In *Proceedings of the Ninth European Conference on Machine Learning*, 1997.

[15] T.R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, et al. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, vol. 286, no. 5439 pp. 531-537, 1999.

[16] Martin Gutlein. Large scale attribute selection using wrappers. Masters' thesis, Albert-Ludwigs-Universitat, Freiburg, 2006.

[17] M. A. Hall and L. A . Smith. Feature subset selection: A correlation based filter approach. In *Proceedings of the Fourth International Conference on Neural Information Processing and Intelligent Information Systems*, pp. 855-858, Dunedin, New Zealand, 1997.

[18] M.A. Hall. Correlation-based feature subset selection for machine learning. PhD thesis, Department of Computer Science, University of Waikato, Waikato, New Zealand, 1999.

[19] Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1998.

[20] G. John, R. Kohavi and K. Pfleger. Irrelevant features and the subset selection problem. In *Proceedings of the Eleventh International Conference on Machine Learning*, pp. 121-129, New Brunswick, NJ, Morgan Kaufmann, 1994.

[21] Kenji Kira and Larry A. Rendell. A practical approach to feature selection. In *Proceedings of the Ninth International Conference on Machine Learning*, 1992.

[22] R. Kohavi. Feature subset selection as search with probabilistic estimates. In *AAAI Fall Symposium on Relevance*, 1994.

[23] R. Kohavi and D. Sommerfield. Feature subset selection using the wrapper method: Overfitting and dynamic search space topology. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, AAAI Press, 1995.

[24] R. Kohavi and G.H. John. Wrappers for feature subset selection. *Artificial Intelligence*, vol. 97, no. 1-2, pp. 273 324, 1997.

[25] Daphne Koller and Mehran Sahami. Toward optimal feature selection. In *Proceedings of the Thirteenth International Conference on Machine Learning*, Morgan Kaufmann, 1996.

[26] I. Kononenko. Estimating attributes: analysis and extension of relief. In *Proceedings of European Conference on Machine Learning*, 1994.

[27] P. Langley. Selection of relevant features in machine learning. In *Proceedings of the AAAI Fall Symposium on Relevance*, AAAI Press, 1996.

[28] P. Langley and W. Iba. Average-case analysis of a nearest neighbor algorithm. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, pp. 889-894, Chambery, France, 1993.

[29] H. Liu and R. Setiono. Feature selection and classification – A probabilistc wrapper approach. In *Proceedings of the Ninth International Conference on Industrial and Engineering Applications of AI and ES*, 1996.

[30] H. Liu and R. Setiono. A probabilistic approach to feature selection: A filter solution. In *Proceedings of the Thirteenth International Conference on Machine Learning*, Morgan Kaufmann, 1996.

[31] T.M. Mitchell. Machine Learning. McGraw-Hill Higher Education, 1997.

[32] Andrew Y. Ng. On feature selection: Learning with exponentially many irrelevant features as training examples. In *ICML*, pp. 404-412, 1998.

[33] David Opitz. Feature selection for ensembles. *AAAI/IAAI*, pp. 379-384, 1999.

[34] S.L. Pomeroy, P. Tamayo, M. Gaasenbeek, L.M. Sturla, M. Angelo, M.E. McLaughlin, J.Y.H. Kim, L.C. Goumnerova, P.M. Black, C. Lau, et al. Prediction of central nervous system embryonal tumor outcome based on gene expression. *Nature*, vol. 415, no. 6870, pp. 436-442, 2002.

[35] J. R. Quinlan, C4.5: Programs for machine learning. Morgan Kaufmann, Los Altos, California, 1993.

[36] Robert E. Schapire. The strength of weak learnability. *Machine Learning*, vol. 5, pp. 197-227, 1990.

[37] David J. Stracuzzi and Paul E. Utgoff. Randomized variable elimination. *J. Mach. Learn. Res.*, 5:1331 1362, 2004. ISSN 1533-7928.

[38] A. Tsymbal, S. Puuronen, D. Patterson. Ensemble feature selection with the simple Bayesian classification. *Information Fusion*, Elsevier Science vol. 4, no. 2, pp. 87-100, 2003.

[39] W.H. Witten and E. Frank. Data Mining: Practical machine learning tools and techniques with java implementation (2 Ed.) Morgan Kaufmann, San Francisco, CA, 2005.

[40] Wikipedia. Occam's Razor [http://en.wikipedia.org/wiki/Occam's_Razor]. 2007.

[41] Jihoon Yang and Vasant Honavar. Feature subset selection using a genetic algorithm. *IEEE Intelligent Systems*, 1998

[42] Zijian Zheng and Geoffrey I. Webb. Stochastic attribute selection committees. *Australian Joint Conference on Artificial Intelligence*, pp. 321-332, 1998.

[43] Zijian Zheng and Geoffrey I. Webb. Stochastic attribute selection committees with multiple boosting: Learning more accurate and more stable classifier committees," *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 123-132, 1999.

Appendix A

Linear Sequential Selection

Dataset	A	B	C	D	E
credit-rating	77.86	85.43	85.51	85.29	85.25
labor	93.57	85.40	81.47	82.57	84.50
vote	90.02	95.63	95.63	95.45	95.33
primary-tumor	49.71	44.51	43.52	44.40	44.52
lymphography	83.13	80.12	75.83	75.49	77.91
vehicle	44.68	53.15	40.87	42.86	51.42
hepatitis	83.81	82.83	80.13	81.68	82.35
segment	80.17	89.15	71.46	81.23	89.44
german_credit	75.16	73.95	73.85	73.34	73.90
horse-colic	78.70	83.78	82.59	83.67	84.19
autos	57.41	63.14	56.71	61.90	63.62
horse-colic.ORIG	66.18	73.42	67.50	73.34	72.93
ionosphere	82.17	90.32	87.38	89.13	90.20
soybean	92.94	92.42	87.15	88.68	90.16
kr-vs-kp	87.79	94.34	90.43	94.11	94.11
anneal	86.59	90.27	91.58	91.62	92.30
anneal.ORIG	75.03	87.62	78.07	81.86	85.86
sonar	67.71	72.84	73.23	71.30	71.96
audiology	72.64	75.64	68.49	68.74	75.91
arrhythmia	62.40	69.26	67.37	68.21	67.92
colon-cancer	55.69	81.05	77.74	82.05	83.07
lymphoma	92.30	83.65	89.60	91.50	90.40
CNS	61.00	53.83	55.00	57.50	57.00
leukemia	91.83	93.92	96.42	98.00	98.25

Table A.1: The classification accuracy of naive Bayes without attribute selection, using GreedyStepwise and using *Linear Sequential Selection*.

A = naive Bayes, no attr. selection

B = naive Bayes, GreedyStepwise

C = naive Bayes, LSS, K = 2

D = naive Bayes, LSS, K = 5

E = naive Bayes, LSS, K = 10

Dataset	A	B	C	D
credit-rating	3.73	1.00	1.35	3.59
labor	3.37	5.07	4.78	3.15
vote	2.78	1.00	1.99	2.04
primary-tumor	10.38	8.50	10.06	10.32
lymphography	4.26	2.58	4.41	5.93
vehicle	2.90	1.23	1.55	6.50
hepatitis	3.39	2.59	2.50	2.86
segment	6.95	2.99	5.30	7.03
german_credit	6.69	4.22	5.39	6.97
horse-colic	3.73	3.79	2.71	3.10
autos	6.62	4.33	5.87	6.29
horse-colic.ORIG	4.01	1.09	1.90	2.64
ionosphere	6.52	3.31	4.98	5.70
soybean	14.12	11.75	13.39	12.26
kr-vs-kp	5.00	3.00	4.93	5.01
anneal	7.95	4.70	4.57	8.21
anneal.ORIG	5.18	1.00	1.96	2.63
sonar	4.97	1.76	2.20	2.76
audiology	8.01	4.86	6.20	7.62
arrhythmia	11.89	4.60	5.17	5.84
colon-cancer	4.80	2.49	1.81	2.24
lymphoma	3.18	2.16	2.21	2.63
CNS	5.04	2.69	3.37	3.87
leukemia	1.17	1.26	1.19	1.21

Table A.2: The number of attributes selected by GreedyStepwise and *Linear Sequential Selection*.

A = naive Bayes, GreedyStepwise

B = naive Bayes, LSS, K = 2

C = naive Bayes, LSS, K = 5

D = naive Bayes, LSS, K = 10

Dataset	A	B	C	D
credit-rating	972.35	147.18	142.81	2254.65
labor	166.09	93.66	93.10	302.56
vote	320.09	79.04	70.29	384.81
primary-tumor	1637.88	903.06	702.42	3000.36
lymphography	464.41	88.99	177.03	1260.78
vehicle	2321.22	515.31	398.31	13841.19
hepatitis	366.41	97.68	85.14	512.24
segment	28164.48	4617.55	7989.44	63052.91
german_credit	3025.48	618.99	689.27	5802.81
horse-colic	718.18	169.17	127.86	826.56
autos	2388.40	462.99	551.57	3453.91
horse-colic.ORIG	1071.16	71.42	105.44	778.83
ionosphere	5486.85	388.05	650.33	3909.23
soybean	16103.89	2780.12	2350.80	10220.09
kr-vs-kp	7504.85	1057.43	1169.93	10918.75
anneal	12224.13	1413.80	1069.91	21420.71
anneal.ORIG	5739.98	440.69	554.67	4851.04
sonar	4719.06	212.12	155.09	1057.59
audiology	5027.25	211.48	339.71	2263.13
arrhythmia	418901.23	2310.19	2367.55	19491.13
colon-cancer	56100.81	202.66	231.74	302.04
lymphoma	80427.64	285.34	319.72	417.08
CNS	375779.45	698.75	819.94	1045.80
leukemia	99694.15	440.87	484.40	551.13

Table A.3: The runtime of naive Bayes using GreedyStepwise and using *Linear Sequential Selection*.

A = naive Bayes, GreedyStepwise

B = naive Bayes, LSS, K = 2

C = naive Bayes, LSS, K = 5

D = naive Bayes, LSS, K = 10

Dataset	A	B	C	D	E
credit-rating	85.57	85.35	85.51	85.51	85.48
labor	78.60	76.13	78.30	78.90	76.83
vote	96.57	95.63	95.63	95.63	95.63
primary-tumor	41.39	39.45	37.32	40.75	39.57
lymphography	75.84	77.61	76.70	76.88	76.20
vehicle	72.28	69.98	64.47	67.09	69.91
hepatitis	79.22	82.29	82.08	82.67	82.15
segment	96.79	96.75	93.62	96.31	96.55
german_credit	71.25	70.99	72.49	71.79	71.63
horse-colic	85.16	84.20	81.49	84.48	84.37
autos	81.77	72.95	73.87	73.45	72.96
horse-colic.ORIG	66.31	66.31	66.31	66.31	66.31
ionosphere	89.74	92.73	90.20	90.03	90.94
soybean	91.78	91.52	86.98	88.52	90.25
kr-vs-kp	99.44	94.34	90.43	94.04	94.11
anneal	98.57	98.56	97.29	98.09	98.49
anneal.ORIG	92.35	91.30	91.60	91.31	91.36
sonar	73.61	72.53	71.73	71.20	70.81
audiology	77.26	76.07	66.77	69.60	76.52
arrhythmia	65.65	69.34	64.65	65.40	66.80
colon-cancer	81.95	77.45	78.50	78.19	79.33
lymphoma	75.50	80.90	78.60	79.45	78.75
CNS	59.50	64.67	66.00	69.33	68.17
leukemia	88.00	87.50	89.00	89.00	89.00

Table A.4: The classification accuracy of J48 without attribute selection, using GreedyStepwise and using *Linear Sequential Selection*.

A = J48, no attr. selection

B = J48, GreedyStepwise

C = J48, LSS, K = 2

D = J48, LSS, K = 5

E = J48, LSS, K = 10

Dataset	A	B	C	D
credit-rating	1.22	1.00	1.00	1.07
labor	2.41	1.72	2.15	2.37
vote	1.00	1.00	1.00	1.00
primary-tumor	8.00	5.09	7.14	7.93
lymphography	3.61	2.43	3.24	4.31
vehicle	7.77	6.46	6.71	8.56
hepatitis	1.89	1.62	1.54	1.73
segment	6.87	7.50	7.15	6.71
german_credit	3.84	4.03	4.48	4.61
horse-colic	3.15	1.13	2.03	2.34
autos	5.45	3.71	4.25	4.84
horse-colic.ORIG	0.00	1.00	1.00	1.00
ionosphere	5.02	2.92	3.49	4.54
soybean	14.47	13.92	13.53	12.40
kr-vs-kp	5.00	3.00	4.12	4.28
anneal	8.59	5.91	7.94	8.45
anneal.ORIG	7.77	6.37	6.95	7.90
sonar	6.40	1.11	1.44	2.55
audiology	10.08	4.31	5.31	9.65
arrhythmia	9.09	4.44	5.15	6.04
colon-cancer	3.15	1.34	1.69	2.13
lymphoma	1.36	1.10	1.23	1.31
CNS	4.01	1.38	1.53	1.82
leukemia	1.14	1.00	1.02	1.13

Table A.5: The number of attributes selected by GreedyStepwise and *Linear Sequential Selection*.

A = J48, GreedyStepwise

B = J48, LSS, K = 2

C = J48, LSS, K = 5

D = J48, LSS, K = 10

Dataset	A	B	C	D
credit-rating	698.00	68.09	165.42	361.95
labor	299.97	37.81	99.08	199.91
vote	171.90	26.50	62.99	105.09
primary-tumor	3637.98	324.04	1368.79	2995.30
lymphography	621.01	74.16	225.31	549.22
vehicle	40640.74	7178.78	17309.10	44229.71
hepatitis	447.61	65.43	106.77	222.93
segment	137441.11	31516.95	65192.51	111425.01
german_credit	8140.34	921.30	2882.59	5518.88
horse-colic	1712.67	30.65	161.15	435.47
autos	12529.19	525.53	1834.58	4696.05
horse-colic.ORIG	156.78	54.45	118.10	193.34
ionosphere	26556.45	826.89	2549.35	7183.28
soybean	83440.94	3647.88	9357.65	18008.29
kr-vs-kp	15226.66	518.30	1852.93	3800.79
anneal	44158.16	1447.22	5218.48	12882.20
anneal.ORIG	81522.66	4192.41	11513.29	29413.47
sonar	47017.64	118.36	357.97	1606.72
audiology	23002.08	210.36	713.17	3614.23
arrhythmia	1154406.61	3986.37	10276.91	25996.16
colon-cancer	82053.84	217.57	306.31	508.32
lymphoma	54938.90	305.26	336.81	426.31
CNS	566855.86	690.35	803.03	1018.05
leukemia	103706.44	446.28	469.16	565.39

Table A.6: The runtime of J48 using GreedyStepwise and using *Linear Sequential Selection*.

A = J48, GreedyStepwise

B = J48, LSS, K = 2

C = J48, LSS, K = 5

D = J48, LSS, K = 10

Dataset	A	B	C	D	E
credit-rating	81.57	85.26	85.51	85.51	85.33
labor	84.30	80.13	78.50	83.30	77.57
vote	92.58	94.80	95.63	95.45	94.99
primary-tumor	39.91	39.35	40.09	40.54	40.15
lymphography	81.69	76.73	74.12	76.94	76.46
vehicle	69.59	68.27	61.94	69.47	68.23
hepatitis	81.40	83.71	80.78	82.56	83.27
segment	97.15	96.93	93.47	96.99	96.98
german_credit	71.88	72.26	71.49	71.22	71.23
horse-colic	79.11	83.72	82.22	83.96	84.12
autos	74.55	78.17	78.26	79.46	79.55
horse-colic.ORIG	65.18	76.16	67.34	67.04	67.76
ionosphere	87.10	88.83	88.97	89.34	89.32
soybean	91.20	93.21	88.17	89.75	92.12
kr-vs-kp	96.12	94.34	90.43	94.41	94.76
anneal	99.13	99.55	98.69	98.98	99.53
anneal.ORIG	95.49	97.88	93.99	97.07	97.62
sonar	86.17	77.89	68.19	67.52	72.29
audiology	78.43	75.34	67.39	68.98	75.51
arrhythmia	53.20	61.51	56.66	57.59	57.52
colon-cancer	76.83	71.55	70.55	73.21	74.14
lymphoma	75.20	80.79	80.55	81.75	83.20
CNS	59.50	56.00	54.67	51.00	54.00
leukemia	78.00	85.86	86.58	86.58	86.58

Table A.7: The classification accuracy of IBk without attribute selection, using GreedyStepwise and using *Linear Sequential Selection*.

A = IBk, no attr. selection

B = IBk, GreedyStepwise

C = IBk, LSS, K = 2

D = IBk, LSS, K = 5

E = IBk, LSS, K = 10

Dataset	A	B	C	D
credit-rating	3.82	1.00	1.00	3.65
labor	4.24	3.21	3.95	3.90
vote	3.16	1.00	1.99	2.98
primary-tumor	7.52	5.67	6.27	7.31
lymphography	4.82	2.58	4.77	5.65
vehicle	6.96	5.95	8.49	6.51
hepatitis	2.30	1.63	1.36	1.74
segment	5.08	6.38	6.58	5.62
german_credit	3.51	2.60	2.84	3.53
horse-colic	2.75	2.71	2.15	2.59
autos	4.42	3.98	4.57	4.48
horse-colic.ORIG	3.71	1.14	1.98	3.54
ionosphere	5.18	3.64	4.06	4.36
soybean	16.16	10.63	13.81	16.63
kr-vs-kp	5.00	3.00	5.12	6.26
anneal	8.24	5.48	6.77	7.73
anneal.ORIG	9.34	6.47	8.42	9.11
sonar	7.84	2.41	4.01	6.81
audiology	13.71	3.60	4.25	9.51
arrhythmia	10.98	1.00	1.00	1.21
colon-cancer	4.61	2.80	3.18	3.47
lymphoma	2.92	2.55	2.92	3.35
CNS	3.35	2.02	2.76	3.40
leukemia	1.67	1.47	1.67	1.70

Table A.8: The number of attributes selected by GreedyStepwise and *Linear Sequential Selection*.

A = IBk, GreedyStepwise

B = IBk, LSS, K = 2

C = IBk, LSS, K = 5

D = IBk, LSS, K = 10

Dataset	A	B	C	D
credit-rating	17278.91	982.96	2378.09	14693.09
labor	638.39	76.95	246.79	563.11
vote	4599.04	374.13	1672.25	5269.08
primary-tumor	8270.84	1324.32	3626.05	9704.04
lymphography	1909.65	170.77	759.99	1857.48
vehicle	29907.92	5138.57	19734.19	31668.24
hepatitis	1170.42	139.74	262.11	640.14
segment	212611.40	38830.42	102643.26	178174.96
german_credit	36771.97	3823.32	9612.88	22611.31
horse-colic	8578.40	1078.13	1715.80	4346.63
autos	4054.21	378.52	1061.17	2044.79
horse-colic.ORIG	21015.16	509.34	1811.76	7598.77
ionosphere	18737.06	871.22	2285.01	4840.36
soybean	257026.08	11499.79	46835.42	127085.87
kr-vs-kp	825462.20	35797.99	153136.89	388848.57
anneal	182538.71	7462.12	25128.61	58321.89
anneal.ORIG	885630.91	16578.78	79812.56	240455.08
sonar	17294.44	256.43	1199.00	4334.02
audiology	65133.18	371.20	1059.73	7069.20
arrhythmia	539309.23	569.99	1164.46	2464.26
colon-cancer	84264.62	273.61	361.99	560.44
lymphoma	135202.32	384.97	437.78	598.30
CNS	706400.17	790.30	954.96	1217.04
leukemia	171110.95	518.88	578.78	652.97

Table A.9: The runtime of IBk using GreedyStepwise and using *Linear Sequential Selection*.

A = IBk, GreedyStepwise

B = IBk, LSS, K = 2

C = IBk, LSS, K = 5

D = IBk, LSS, K = 10

Appendix B

Randomized Linear Sequential Selection

Dataset	A	B	C	D	E	F	G
credit-rating	85.48	65.67	85.20	85.48	85.54	85.36	85.25
labor	83.70	80.73	81.87	80.90	83.77	82.33	81.40
vote	95.63	82.32	95.45	95.93	95.38	95.08	95.31
primary-tumor	33.27	31.39	44.84	41.92	44.61	44.78	44.49
lymphography	76.36	68.48	75.55	78.86	77.57	78.38	77.99
vehicle	45.43	50.78	48.76	48.90	53.80	53.32	53.22
hepatitis	81.75	84.10	81.63	80.00	82.42	82.34	81.56
segment	63.83	76.24	75.00	71.42	89.23	89.23	78.84
german_credit	73.52	70.02	73.24	73.45	73.61	73.45	73.52
horse-colic	80.51	68.85	82.91	81.39	84.19	83.72	83.64
autos	51.81	45.81	59.69	59.82	62.80	62.90	63.52
horse-colic.ORIG	68.72	65.73	73.23	68.13	72.71	72.96	73.01
ionosphere	87.21	76.95	89.55	89.83	90.17	90.01	90.46
soybean	70.62	64.62	91.01	89.11	91.66	91.42	92.42
kr-vs-kp	90.59	57.63	94.07	90.43	94.11	94.24	94.24
anneal	83.62	76.58	90.06	85.46	92.36	91.65	90.97
anneal.ORIG	76.17	76.17	78.16	78.64	85.74	86.22	80.41
sonar	73.36	61.05	71.41	72.74	71.20	71.59	71.20
audiology	56.23	28.27	68.73	67.92	75.24	72.54	68.74
arrhythmia	65.75	54.82	68.41	67.70	67.50	68.04	68.49
colon-cancer	70.98	61.95	79.86	76.90	82.76	83.10	81.55
lymphoma	88.75	56.50	90.70	89.60	90.60	90.65	91.25
CNS	59.67	61.17	55.83	56.50	57.00	57.50	56.67
leukemia	93.83	57.92	98.00	96.42	98.25	98.00	98.00

Table B.1: The classification accuracy of naive Bayes using *Randomized Linear Sequential Selection*.

A = naive Bayes, RLSS, K = 2, M = 1

B = naive Bayes, RLSS, K = 2, M = 2

C = naive Bayes, RLSS, K = 5, M = 1

D = naive Bayes, RLSS, K = 5, M = 3

E = naive Bayes, RLSS, K = 10, M = 1

F = naive Bayes, RLSS, K = 10, M = 2

G = naive Bayes, RLSS, K = 10, M = 5

Dataset	A	B	C	D	E	F	G
credit-rating	85.43	62.14	85.41	85.36	85.48	85.48	85.48
labor	81.53	76.60	79.57	78.10	76.83	77.20	78.23
vote	95.63	82.99	95.63	95.63	95.63	95.63	95.63
primary-tumor	30.62	28.59	41.09	38.61	39.39	39.71	39.68
lymphography	77.15	65.68	76.87	77.29	76.55	76.32	77.12
vehicle	66.36	69.40	70.38	70.47	70.29	70.19	70.77
hepatitis	81.59	82.35	82.61	81.77	82.16	82.30	82.55
segment	90.50	93.90	96.16	93.09	96.50	96.73	96.71
german_credit	72.10	70.00	72.18	71.69	71.78	71.51	71.68
horse-colic	80.73	63.84	84.53	81.41	84.29	84.20	84.23
autos	73.08	60.36	73.00	74.47	73.10	72.85	73.28
horse-colic.ORIG	66.31	66.31	66.31	66.31	66.31	66.31	66.31
ionosphere	89.62	85.54	90.51	90.62	90.68	91.00	91.28
soybean	75.01	66.53	90.34	85.87	90.85	91.07	90.98
kr-vs-kp	90.66	59.40	94.05	90.43	94.11	94.24	94.25
anneal	88.12	76.49	97.60	96.95	98.46	98.43	98.02
anneal.ORIG	91.78	76.17	91.17	91.60	91.38	91.35	91.31
sonar	71.43	61.82	71.39	71.54	70.81	70.75	70.95
audiology	60.87	27.08	68.95	67.48	75.36	72.97	69.25
arrhythmia	64.01	54.14	64.94	64.74	66.38	65.21	65.56
colon-cancer	77.93	65.67	78.19	78.50	79.33	78.21	78.05
lymphoma	76.40	47.40	80.20	78.60	79.45	78.95	79.45
CNS	63.50	65.00	69.67	66.00	69.17	69.33	68.50
leukemia	88.67	60.67	89.00	89.00	89.00	89.00	89.00

Table B.2: The classification accuracy of J48 using *Randomized Linear Sequential Selection*.

A = J48, RLSS, K = 2, M = 1

B = J48, RLSS, K = 2, M = 2

C = J48, RLSS, K = 5, M = 1

D = J48, RLSS, K = 5, M = 3

E = J48, RLSS, K = 10, M = 1

F = J48, RLSS, K = 10, M = 2

G = J48, RLSS, K = 10, M = 5

Dataset	A	B	C	D	E	F	G
credit-rating	85.10	58.68	85.16	85.62	85.39	85.38	85.33
labor	78.23	77.93	81.93	78.27	79.17	82.13	82.20
vote	95.63	84.23	95.45	95.13	95.38	94.94	94.99
primary-tumor	31.10	29.94	40.59	39.56	40.13	40.04	40.12
lymphography	73.29	64.50	76.72	76.13	77.50	76.51	76.40
vehicle	61.89	67.16	71.41	66.29	67.62	67.37	70.58
hepatitis	81.39	81.97	82.75	81.97	83.26	82.37	82.56
segment	90.76	95.98	96.79	94.19	96.92	97.03	97.18
german_credit	71.04	69.94	71.31	71.51	71.03	70.65	71.65
horse-colic	80.73	60.66	84.21	81.79	84.12	83.99	83.80
autos	78.72	63.11	79.48	78.10	79.30	79.12	78.58
horse-colic.ORIG	68.64	67.05	67.29	70.35	68.18	69.60	70.22
ionosphere	88.14	84.40	88.88	89.15	89.75	89.35	89.26
soybean	73.42	68.12	93.44	84.35	94.07	93.51	93.93
kr-vs-kp	90.74	59.81	94.10	90.43	94.93	94.76	94.59
anneal	96.33	76.60	98.66	98.69	99.44	99.35	98.91
anneal.ORIG	92.41	76.68	96.23	94.29	97.65	97.38	97.00
sonar	69.72	66.08	70.35	73.81	73.62	75.72	75.34
audiology	54.39	27.60	69.02	69.39	74.94	72.90	69.11
arrhythmia	56.04	54.07	57.57	56.51	57.66	57.46	57.59
colon-cancer	70.24	59.74	72.40	68.71	73.83	71.33	73.38
lymphoma	76.00	54.25	83.85	80.70	83.30	82.15	80.85
CNS	58.50	55.67	53.00	53.67	53.17	54.33	52.17
leukemia	87.83	59.33	86.58	86.33	86.92	86.92	86.58

Table B.3: The classification accuracy of IBk using *Randomized Linear Sequential Selection*.

A = IBk, RLSS, K = 2, M = 1

B = IBk, RLSS, K = 2, M = 2

C = IBk, RLSS, K = 5, M = 1

D = IBk, RLSS, K = 5, M = 3

E = IBk, RLSS, K = 10, M = 1

F = IBk, RLSS, K = 10, M = 2

G = IBk, RLSS, K = 10, M = 5

Appendix C

Feature Selection Ensemble

Dataset	**A**	**B**	**C**
credit-rating	77.78	77.93	78.13
labor	89.50	90.60	89.93
vote	89.31	88.97	88.65
primary-tumor	31.80	34.35	36.28
lymphography	78.58	80.19	80.05
vehicle	46.75	46.42	45.88
hepatitis	79.31	79.38	79.25
segment	83.55	80.73	80.48
german_credit	70.00	70.02	70.04
horse-colic	75.52	77.15	79.14
autos	55.60	56.65	57.39
horse-colic.ORIG	66.31	66.41	66.33
ionosphere	84.59	84.02	82.93
soybean	86.83	87.94	89.78
anneal	76.17	76.17	76.17
anneal.ORIG	76.17	76.17	76.17
sonar	68.72	69.64	71.38
colon-cancer	81.24	82.98	83.55
lymphoma	93.15	93.10	92.90
CNS	64.67	65.00	65.50
leukemia	94.00	93.67	94.00

Table C.1: The classification accuracy of *Feature Selection Ensemble* using naive Bayes.

A = naive Bayes, FSE, LSS, K = 2

B = naive Bayes, FSE, LSS, K = 5

C = naive Bayes, FSE, LSS, K = 10

Dataset	**A**	**B**	**C**
credit-rating	82.35	82.93	83.04
labor	66.40	68.23	68.40
vote	90.07	90.72	90.60
primary-tumor	25.22	27.79	29.71
lymphography	76.26	77.74	78.23
vehicle	70.77	71.17	70.42
hepatitis	79.38	79.38	79.44
segment	93.56	93.89	94.35
german_credit	70.00	70.00	70.00
horse-colic	67.28	69.79	70.19
autos	74.10	75.36	74.30
horse-colic.ORIG	66.31	66.31	66.31
ionosphere	90.12	90.17	90.60
soybean	87.29	88.24	91.76
anneal	76.19	76.18	76.17
anneal.ORIG	76.17	76.17	76.17
sonar	75.30	76.77	78.19
colon-cancer	83.14	84.76	83.48
lymphoma	91.10	90.45	90.25
CNS	65.83	65.50	63.83
leukemia	93.67	93.67	93.67

Table C.2: The classification accuracy of *Feature Selection Ensemble* using J48.

A = J48, FSE, LSS, K = 2

B = J48, FSE, LSS, K = 5

C = J48, FSE, LSS, K = 10

Dataset	A	B	C
credit-rating	79.25	79.17	78.55
labor	88.40	87.87	87.33
vote	90.69	92.28	92.07
primary-tumor	29.06	33.19	33.95
lymphography	79.93	79.73	79.13
vehicle	67.78	68.93	69.48
hepatitis	79.56	79.89	79.96
segment	93.35	94.51	95.13
german_credit	69.98	70.00	70.06
horse-colic	73.02	73.43	75.58
autos	79.48	78.69	77.95
horse-colic.ORIG	68.59	68.07	68.27
ionosphere	91.97	92.14	92.54
soybean	88.89	90.22	91.43
anneal	76.74	76.35	76.29
anneal.ORIG	76.17	76.17	76.17
sonar	77.17	79.11	80.61
colon-cancer	82.14	82.40	82.60
lymphoma	92.25	92.55	92.10
CNS	60.17	58.33	60.17
leukemia	93.17	93.42	92.00

Table C.3: The classification accuracy of *Feature Selection Ensemble* using IBk.

A = IBk, FSE, LSS, K = 2

B = IBk, FSE, LSS, K = 5

C = IBk, FSE, LSS, K = 10

Appendix D

Random Feature Ensemble

Dataset	A	B	C	D
anneal	92.92	93.08	92.99	93.09
anneal.ORIG	82.32	82.56	82.43	82.56
autos	64.73	65.27	64.29	65.12
horse-colic	83.56	83.53	83.64	83.53
horse-colic.ORIG	73.56	73.50	73.34	73.45
credit-rating	85.25	85.25	85.25	85.25
german_credit	73.66	73.80	73.86	73.77
hepatitis	81.75	81.82	81.75	81.75
ionosphere	91.43	91.40	91.37	91.40
labor	81.40	81.40	81.40	81.40
lymphography	79.01	78.48	78.81	78.40
primary-tumor	44.49	44.49	44.49	44.49
segment	83.33	83.55	83.50	83.57
sonar	72.60	72.21	72.26	72.21
soybean	92.75	93.00	92.96	93.00
vehicle	55.17	54.93	55.00	55.17
vote	95.31	95.31	95.31	95.31
vowel	67.15	67.15	67.15	67.15

Table D.1: The classification accuracy of *Random Feature Ensemble* using naive Bayes.

A = naive Bayes, RFE, RLSS, K = 10, M = 5, I = 10

B = naive Bayes, RFE, RLSS, K = 10, M = 5, I = 30

C = naive Bayes, RFE, RLSS, K = 10, M = 5, I = 50

D = naive Bayes, RFE, RLSS, K = 10, M = 5, I = 100

Dataset	A	B	C	D
anneal	98.13	98.14	98.13	98.13
anneal.ORIG	91.37	91.31	91.33	91.31
autos	74.47	74.53	74.33	74.24
horse-colic	84.31	84.29	84.18	84.23
horse-colic.ORIG	66.31	66.31	66.31	66.31
credit-rating	85.48	85.48	85.48	85.48
german_credit	71.75	71.77	71.72	71.77
hepatitis	82.55	82.55	82.55	82.55
ionosphere	92.05	92.22	92.00	92.02
labor	78.03	78.03	78.23	78.03
lymphography	77.80	77.81	77.80	77.94
primary-tumor	39.68	39.68	39.68	39.68
segment	96.73	96.67	96.76	96.74
sonar	71.96	71.72	71.72	71.77
soybean	92.63	92.80	92.58	92.74
vehicle	70.72	70.70	70.72	70.92
vote	95.63	95.63	95.63	95.63
vowel	81.85	81.85	81.85	81.85

Table D.2: The classification accuracy of *Random Feature Ensemble* using J48.

A = J48, RFE, RLSS, K = 10, M = 5, I = 10

B = J48, RFE, RLSS, K = 10, M = 5, I = 30

C = J48, RFE, RLSS, K = 10, M = 5, I = 50

D = J48, RFE, RLSS, K = 10, M = 5, I = 100

Dataset	**A**	**B**	**C**	**D**
anneal	99.01	99.00	99.01	99.01
anneal.ORIG	97.26	97.22	97.19	97.17
autos	79.12	79.11	79.31	79.22
horse-colic	83.83	83.74	83.77	83.77
horse-colic.ORIG	74.18	74.66	74.90	74.64
credit-rating	85.33	85.33	85.33	85.33
german_credit	71.92	71.75	71.78	71.75
hepatitis	82.68	82.68	82.75	82.75
ionosphere	90.22	90.20	90.54	90.17
labor	82.53	82.20	82.20	82.20
lymphography	77.83	77.83	77.83	77.83
primary-tumor	40.12	40.12	40.12	40.12
segment	97.25	97.22	97.24	97.25
sonar	80.55	80.59	81.55	81.55
soybean	94.92	95.10	95.10	95.10
vehicle	70.65	70.65	70.65	70.65
vote	94.99	94.99	94.99	94.99
vowel	98.79	98.79	98.80	98.80

Table D.3: The classification accuracy of *Random Feature Ensemble* using IBk.

A = IBk, RFE, RLSS, K = 10, M = 5, I = 10

B = IBk, RFE, RLSS, K = 10, M = 5, I = 30

C = IBk, RFE, RLSS, K = 10, M = 5, I = 50

D = IBk, RFE, RLSS, K = 10, M = 5, I = 100